情報処理学会論文誌 Vol.61 No.9 1414–1427 (Sep. 2020)

マルウェア検体のデータ欠損がアンチウイルスによるマルウェア同定に与える影響の調査

小久保 博崇[1,a)] 大山 恵弘[2]

受付日 2019年11月25日, 採録日 2020年6月1日

概要：マルウェアを使った攻撃を受けたとき，どのようなマルウェアが攻撃に使われたかを知ることは被害状況を知るためにも重要である．しかし，マルウェアの中には証拠隠滅や調査妨害などの目的で，自分自身を消去するものも存在する．消去されてしまったマルウェアはデジタルフォレンジック技術によってある程度復元することが可能な場合もあるが，マルウェアに限らず1度消去されたファイルはストレージの使用とともにデータの一部や全部が欠損していく．本論文ではこのようなファイル消去によりデータの一部が欠損したマルウェアに着目し，欠損した状態でどこまでマルウェアを同定できるのか，マルウェアのデータのうちどの部分が欠損すると同定に影響が出るのかについて調査を行った．NTFS におけるファイルの欠損の仕方をもとに人工的にマルウェアを欠損させ調査したところ，特にファイル先頭やファイル末尾の欠損がマルウェアの同定や悪性判定の精度を著しく下げることが分かった．

キーワード：マルウェア，データ欠損，アンチウイルス，マルウェア分類，マルウェア検知，コンピュータフォレンジック

Effect of Malware Data Loss on Malware Identification by Antivirus

Hirotaka Kokubo[1,a)] Yoshihiro Oyama[2]

Received: November 25, 2019, Accepted: June 1, 2020

Abstract: It is important to know what kind of malware is used for an attack when it is attacked with malware in order to know the damage situation. However, some types of malware erase themselves for the purpose of destruction of evidence or obstruction of investigation. Erased malware can sometimes be recovered by digital forensic technology. Nevertheless some of the data in the deleted file are lost as the storage is used. In this paper, we focus on the malware which has lost parts of the data by such file deletion. We investigate the accuracy of identification of malware with missing data and which part of the data of the malware affects the identification. We artificially generate corrupted malware by investigating to the file data loss in NTFS and analyze that corrupted malware. It was found that missing the top and end of the file had a significant adverse effect on malware identification.

Keywords: malware, data loss, antivirus, malware classification, malware detection, computer forensics

1. はじめに

サイバー攻撃対策において，攻撃に使われたマルウェアがどんな種類のマルウェアであったのかを同定する作業は，攻撃による被害状況を知るうえで重要である．攻撃に使われたマルウェアを被害端末から入手し，マルウェアの種類を突き止めることで，情報流出や組織内感染の可能性の有無，感染経路の推定，改竄を受けうるファイルの特定，マルウェア除去の方法など，攻撃の事後対応に役立つ情報を手に入れることができる．

しかし，攻撃に使われたマルウェアが被害端末上に残っていない場合が存在する．役目を終えて不要になったマルウェアを，攻撃者や次段階のマルウェア，もしくはマル

[1] 株式会社富士通研究所
FUJITSU LABORATORIES LTD., Kawasaki, Kanagawa 211–8588, Japan
[2] 筑波大学
University of Tsukuba, Tsukuba, Ibaraki 305–8577, Japan
[a)] kokubo.hirotaka@fujitsu.com

ウェア自身が削除する場合である．マルウェアに限らず，削除されてしまったファイルはOS上から行う通常のファイル操作によって取り出すことはできないが，デジタルフォレンジック技術を用いることで復元できる場合がある [1]．Windows環境においてよく使用されるFAT（File Allocation Table）やNTFS（NT File System）などのファイルシステムでは，ファイルを削除するとファイルシステム上に存在するそのファイルのための管理データ領域の削除フラグが立つ．削除フラグが立つと，OS上からそのファイルを参照することができなくなる．しかし，この状態でもファイルのデータ自体はストレージ上に残存している．この残存したデータを集めて結合することで元のファイルを復元することができる．しかし，削除フラグが立ったファイルのデータ領域は空き領域として扱われるため，新しく作られた他のファイルのデータが上書きされることがある．通常，端末の使用にはファイル作成がともなうため，端末使用時間の経過とともに削除済みデータが新規ファイルデータにより上書きされてしまう可能性は高まる [2]．そのため，削除されて間もないマルウェアは完全な状態で復元できる可能性が高いといえるが，削除されてから時間が経過したマルウェアのデータは一部または全部が欠損している可能性がある．マルウェアによる攻撃の被害状況を知るうえでは，データの一部が欠損してしまったマルウェア（以下，欠損マルウェアと呼ぶ）からでも，マルウェアの同定が行えることが望ましい．

そこで本研究では，このような欠損マルウェアに着目する．欠損のないマルウェア検体を様々なファイルオフセットから人工的にデータ欠損させることで欠損マルウェアを作り出し，欠損マルウェアのうち，どのような欠損の仕方をした検体が元のマルウェアと同定できなくなるのかを調査した．対象とするマルウェアはx86-64向けのWindows用PE形式のバイナリプログラムとする．

まず本論文では，現実の環境ではどのような形でデータ欠損が起こるかについて述べる．Windows 10で標準的に使われているNTFS上でのデータ欠損の起こり方について実験を行い調査した結果を基に，現実に生成されうる欠損マルウェアの形式について言及する．また，上記から本論文で扱う欠損マルウェアについて定義する．

次に，実マルウェア検体を欠損させることで生成した欠損マルウェアに対する，アンチウイルスによる検知率やマルウェアの同定の成功率を示し，それらが欠損のさせ方によってがどう変わったかを述べるとともに，なぜそのような結果になったかについて考察を行う．

本論文の貢献は以下のとおりである．

- 検体のデータのうち，どのデータ領域が欠損するとマルウェアの同定に特に影響を与えるかを明らかにした．ファイル先頭の欠損が最も検知や同定の精度を低下させた．次点でファイル末尾の欠損による悪影響が大きかった．
- 影響の大きかった欠損位置において，実際にどのような意味のデータが欠損していたかを明らかにした．PEファイルのメタ情報が格納されているヘッダや，リソース情報が格納されているセクションが欠損していた．
- 欠損が起きたマルウェアのうち，どの程度の割合のマルウェアが同定できなくなるかを明らかにした．同定に用いるアンチウイルス製品や欠損位置，欠損量に依存するものの，最も影響の大きいケースであるファイル先頭からの欠損の場合は98.3%から100%の検体が同定できなくなった．

2. ファイルのデータ欠損

まず，消去されたファイルがどのように欠損するのかを確認するために，下記の実験を行った．実験環境のOSはWindows 10，ファイルシステムはNTFS，セクタサイズは512 bytes，クラスタサイズは4,096 bytes（8セクタ）である．仮想マシンではなく物理マシン上で実験を行った．

(1) ハードディスクストレージ上にターゲットファイルを作成し，まだ欠損していない現時点でのファイルのデータ，ファイルのSHA256ハッシュ値，ストレージ上でのデータ位置・サイズを記録する．
(2) OS上の操作でターゲットファイルを消去する．
(3) ストレージ上に新規ファイルを作成する．ただし，新規ファイルのサイズはセクタサイズの非整数倍かつターゲットファイルのサイズ未満で，データの中身は非ゼロのバイト列とする．
(4) (1)で記録したストレージ上でのデータ位置とサイズを基にターゲットファイルを復元し，SHA256ハッシュ値を計算して元のSHA256ハッシュ値と比較する．SHA256ハッシュ値が同一であった場合は欠損が起きていないということであるため，(3)に戻る．
(5) 復元したターゲットファイルのデータと欠損前のデータを比較し，ターゲットファイルのどの部分がどれだけ欠損したかを確認する．

実験の結果，我々の環境では，データ欠損はクラスタサイズである4,096 bytes単位で起こった．ターゲットファイルのデータ領域において，クラスタサイズの整数倍のファイルオフセットから新規ファイルのデータによる上書きが始まり，新規ファイルの内容を書き込み尽くしたあとはクラスタサイズの整数倍に到達するまで0で埋められた．ごく一部の環境ではクラスタサイズではなくセクタサイズである512 bytes単位での欠損が起こったが，上書き時に起こるゼロ埋めの挙動などは同様であった．

この実験結果から，本研究ではマルウェア検体に対してクラスタサイズの整数倍のファイルオフセット，すなわち$4{,}096 \times N$番地から，クラスタサイズである4,096 bytes分

のデータを値 0 で上書きすることにより欠損マルウェアを生成するものとする．欠損のための上書きに用いるデータは 0 以外にもランダムデータや良性ソフトウェアから抽出したデータなどが考えられるが，今回はまず単純な方式での結果を得たいと考え 0 を選んだ．実環境での欠損では，前述のとおり欠損が起きたタイミングに書き込まれた実ファイルデータで上書きされ，上書きデータ量がクラスタサイズに満たない場合はクラスタ境界までの残り部分が 0 で上書きされる．しかし，実ファイルデータで上書きした場合は，偶然シグネチャとなりうる文字列が混入するなどしてそのデータの内容が実験結果に影響する可能性があるため，今回は実ファイルデータによる上書きは採用しない．全検体を一律に 0 で欠損させることで，検体間での欠損条件の差異をなくすとともに再現を容易にした．実際の環境では連続しない位置にある複数のクラスタが欠損することもあるが，これも単純化のために単一のクラスタまたは連続したクラスタの欠損のみを取り扱う．他にも欠損のさせ方としては，セクションや PE ヘッダといった意味を持って区切られているデータ領域ごとに欠損させることも考えられるが，消去されたファイルを復元したときに起こる自然欠損においてはデータの意味を考慮した欠損は起きないため，本研究では取り扱わない．

3. 実験

3.1 使用する検体

実験に使用する検体群は，我々が独自に収集した x86-64 向けの PE 形式のマルウェア 924 検体である．各検体の MD5 ハッシュ値は 924 検体すべてで異なる．これらの検体群の基本的な情報を**表 1** に示す．

マルウェアの名称の種類数は，分類に使用するアンチウイルス製品（以下，AV と呼ぶ）によって著しく異なり，平均で 89 種，最も多く分類する場合で 547 種に分類された．表 1 のマルウェア名種類数には著名な AV による分類数を記載した．この AV での分類によると，最も多く含まれているマルウェアは Trojan:Win64/Alureon.L で 129 検体，2 番目が Trojan:Win32/Bitrep.A で 21 検体，3 番目が Virus:Win32/Expiro.EM!bit で 15 検体であった．確認できたマルウェア名らのうち約半数のマルウェア名は，検体群中にそれぞれ 1 検体ずつ含まれていた．また，ある AV によってただ 1 種の名前が付けられた検体群も，別の AV によって複数の様々な名前を付けられることがある．たとえば前述の Trojan:Win64/Alureon.L は別のある AV では 3 種の名前が付けられており，また別の AV では 30 種の名前が，さらに別の AV では 1 種の汎用的な名称（Generic 名）が付けられている．

同位置かつ同量の欠損でも，マルウェアのサイズによっては欠損するデータの意味が異なってくるため，各マルウェアのファイルサイズについても言及する．マルウェアのファイルサイズは最小が 36 KiB で，最大が約 1.95 MiB であり，平均は約 387 KiB，標準偏差は約 381 KiB である．全体の 75%の検体のファイルサイズが約 623 KiB 未満である．

表 1 検体情報
Table 1 The details of specimen.

項目	値
検体数	924 検体
マルウェア名種類数	70 種および名称不明 546 検体
サイズの最小値	36,864 bytes
サイズの平均値	396,508.7 bytes
サイズの最大値	2,043,296 bytes
サイズの標準偏差	390,329.6 bytes

3.2 実験手法

3.2.1 実験 A：ファイル先頭付近から 4,096 bytes の欠損

前述の検体群に対して欠損処理を行い，自然な形で欠損したマルウェアを人工的に生成する．検体データのファイルオフセット 0x0000 から 4,096 bytes 分の領域を 0 で上書きしたグループ，ファイルオフセット 0x1000 から 4,096 bytes 分の領域を 0 で上書きしたグループのように，欠損開始ファイルオフセットを 0x0000 から 0x8000 まで 0x1000（4,096 の 16 進数表現）ずつ増やしていき，そのファイルオフセットから 4,096 bytes 分を 0 で欠損させることで，9 つの欠損マルウェアグループ（各グループあたり 924 検体，合計 8,316 検体）を用意した（**表 2**）．欠損処理を行うためには元々の検体のデータサイズが開始ファイルオフセットと 4,096 bytes の合計以上でなければならないが，ファイルサイズが小さく欠損処理を行えない検体はあらかじめ除外しているため，元々の 924 検体には含まれていない．

それぞれのグループに対して VirusTotal[*1] を使用し，各社の AV でのスキャンを行い，結果を分析した．

表 2 欠損マルウェアグループ
Table 2 Corrupted malware groups.

	欠損開始ファイルオフセット	欠損サイズ	検体数
Group 1	0x0000	4,096	924
Group 2	0x1000	4,096	924
Group 3	0x2000	4,096	924
Group 4	0x3000	4,096	924
Group 5	0x4000	4,096	924
Group 6	0x5000	4,096	924
Group 7	0x6000	4,096	924
Group 8	0x7000	4,096	924
Group 9	0x8000	4,096	924
Original	—	—	924

[*1] https://www.virustotal.com/

3.2.2 実験 B：良性ソフトウェアの欠損

欠損処理はファイルを通常とは異なる状態に変える．そのため，良性ソフトウェアであっても欠損によりマルウェアと誤判定される可能性がある．このような影響を調査するために，良性ソフトウェアについても欠損処理を行う．対象とする良性ソフトウェアは，Windows 10 において `%SystemRoot%` フォルダ以下に存在する x86-64 向けの PE 実行ファイル 835 種である．我々の環境では，`%SystemRoot%` は C ドライブ直下の `Windows` フォルダを指す．3.2.1 項と同様に，ファイルサイズが小さく欠損処理を行えない良性ソフトウェアはあらかじめ除外している．該当フォルダ以下には，ファイルパス以外は完全に同一のファイルが複数存在しているケースがある．同一のファイルを対象としてしまうことを避けるために，それぞれの良性ソフトウェアについて MD5 ハッシュ値を算出し，重複がないよう良性ソフトウェアを選定した．これら 835 種の検体をファイルオフセット 0x0000 から 4,096 bytes 分，またはファイルオフセット 0x1000 から 4,096 bytes 分欠損させ，各社の AV でスキャンを行い誤検知状況を分析した．

3.2.3 実験 C：その他の位置からの欠損

先頭付近以外からの欠損パターンの調査として，ファイル末尾からの欠損や 4,096 bytes よりも多い量の欠損について実験を行う．以降では，4,096 bytes 分以外の欠損量も取り扱うため，欠損量に関しても 0x1000 のように 16 進数で表記する．

1 つ目は，ファイル先頭付近から複数クラスタの欠損である．欠損開始ファイルオフセットを 0x0000 から 0x6000 まで 0x1000（1 クラスタサイズ）ずつずらし，それぞれ 0x2000 bytes 分または 0x3000 bytes 分のデータを欠損させる．

2 つ目は，ファイル末尾からクラスタサイズの整数倍分の欠損である．ファイル末尾からファイル先頭方向へ 0x1000，0x2000，0x3000，0x4000，0x5000，0x6000 bytes 分のデータを欠損させる．ファイル末尾を欠損の起点としているため，3.2.1 項や 3.2.2 項と異なり，クラスタ境界を無視した欠損処理を行っている．クラスタ境界を無視した 0x1000 bytes の連続したデータを以後ブロックと呼称する．

3 つ目は，ファイル末尾付近から単一ブロックの欠損である．欠損開始ファイルオフセットをファイル末尾から 0x1000 ずつファイル先頭方向へずらし，それぞれ 1 ブロック分のデータを欠損させる．この実験も 2 つ目と同様に，クラスタ境界を無視した欠損処理を行っている．

また，本項の実験は 3.2.1 項の実験 A の 9 カ月後に実施しているため，実験 A 実施時と比べ AV の検知率が向上している．そのため，実験 A の結果と数値を単純に比較することは行わない．

4. 結果

まず 3.2.1 項の実験 A の結果を，4.1 節および 4.2 節に示す．

4.1 欠損が検知率に与える影響

まずは，マルウェアの欠損が AV の検知率に与える影響について示す．

4.1.1 欠損マルウェアグループごとの検知率

表 3 に各欠損マルウェアグループと元データ（Original）の検知率を記載した．本項での検知率とは，検体を悪性と判定した AV の種類数をその検体の判定に使用した全 AV の種類数で割った値である．この検知率を各グループごとに集計し平均値，最小値，最大値，中央値を算出した．

今回は VirusTotal において使用可能な AV すべてを検知率の計算に使用したが，これら AV は玉石混淆であるため，参考情報として検知率計算に使用する AV を著名な 5 製品に限定した場合の検知率についても**表 4** に記載した．検知率の平均値に関しては全 AV を計算に使用した場合とおおむね同様の傾向を示しているが，欠損前検体の検知率に関してはやや大きく上昇している．

また，**図 1** に各欠損マルウェアグループと元データの検知率（Detection Rate）の分布を箱ひげ図で表現した．ひ

表 3 全 AV での検知率
Table 3 Detection rates (using all antiviruses).

	平均値	最小値	最大値	中央値
Group 1	0.010	0	0.439	0
Group 2	0.356	0	0.761	0.384
Group 3	0.385	0	0.786	0.464
Group 4	0.372	0	0.783	0.386
Group 5	0.397	0	0.789	0.479
Group 6	0.403	0	0.789	0.500
Group 7	0.400	0	0.786	0.496
Group 8	0.396	0	0.775	0.486
Group 9	0.403	0	0.786	0.500
Original	0.478	0	0.851	0.567

表 4 使用 AV を限定した場合の検知率
Table 4 Detection rates (using famous antiviruses).

	平均値	最小値	最大値	中央値
Group 1	0.006	0	0.4	0
Group 2	0.322	0	1	0.4
Group 3	0.322	0	1	0.4
Group 4	0.263	0	1	0.2
Group 5	0.391	0	1	0.4
Group 6	0.400	0	1	0.4
Group 7	0.392	0	1	0.4
Group 8	0.360	0	1	0.4
Group 9	0.392	0	1	0.4
Original	0.610	0	1	0.6

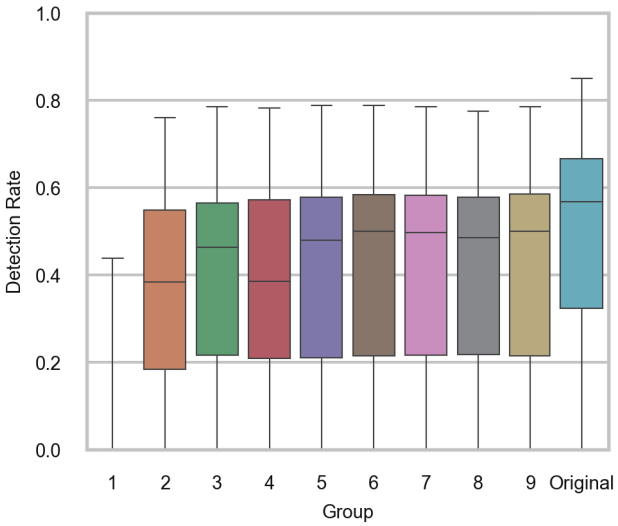

図 1 欠損マルウェア検知率の分布
Fig. 1 Distribution of corrupted malware detection rates.

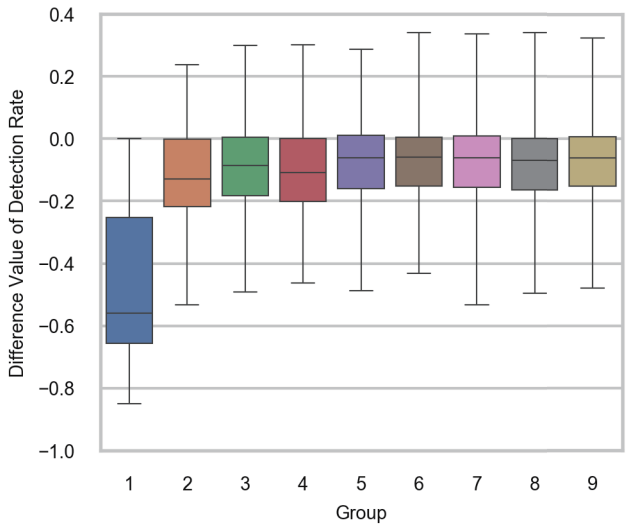

図 2 欠損マルウェア検知率の差分
Fig. 2 Difference value of corrupted malware detection rate.

げの下端が最小値，箱の下辺が 25 パーセンタイル，箱の内部の線が 50 パーセンタイル（中央値），箱の上辺が 75 パーセンタイル，ひげの上端が最大値を表している．これら区切れと区切れの間にはいずれも同量のデータが属しているため，区切れ間の長さが短いほどその区間にデータが集中して出現している．また，箱の中には全体の半数のデータが属している．

すべての Group において，検知率の最小値は 0 であったため，ひげの下端は X 軸と重なっている．Group 1 に関しては，ほとんどの検体の検知率が 0 であるため，箱部分が見えていない．Group 2 と Group 4 は似た分布となっており，中央値の位置が元データに比べて箱の中央付近まで低下している．Group 1，2，4 を除く Group も似た分布になっており，元データの検知率分布がそのまま下方向へシフトした形になっている．

Group 2 以降において，検知率は 0 から約 0.8 まで幅があるが，これは元データに対する検知率を欠損後もある程度引き継いでいるためと考える．Group 1，すなわちファイルオフセット 0x0000 から 0x1000 bytes 分の欠損が，検体の検知率を特に低下させている．中央値で比較すると，Group 2 および Group 4 の検知率が Group 1 に次いで低く，それ以外の Group はあまり差がない．このことから，マルウェアデータのファイルオフセット 0x8000 までから始まるデータ欠損は，場所によらず AV による悪性判定の検知率低下を引き起こしており，特にマルウェア先頭の欠損が致命的であるといえる．また，ファイルオフセット 0x1000 および 0x3000 付近から始まる欠損は，マルウェア先頭からの欠損ほどではないが他の欠損箇所よりも検知率の低下を引き起こす傾向にあるといえる．

図 2 に，各欠損マルウェアグループの検知率から元データの検知率を引いた値（Difference Value of Detection Rate）の分布を箱ひげ図で表現した．元データよりも検知率が低下すると Y 軸の値である差分が 0.0 を下回り，検知率が上昇すると 0.0 を上回る．差分が 0 に等しい場合は検知率が変化しなかったことを表している．Group 1 はすべての検体の検知率が低下している．Group 1 の多くの検体は検知率が 0 まで低下しているため，図 1 の元データのグラフをマイナス方向へ反転させたようなグラフとなっており，グラフの形状にはあまり意味がないといえる．それ以外の Group は 75%程度の検体が検知率の低下を起こしているが，欠損しても検知率が変化しないもしくは，検知率が上昇する検体も合計して 25%ほど存在している．欠損前後で悪性判定に使用する AV の種類を統一したうえで，すべての検体の検知率が低下した Group 1 を除いた各 Group において，欠損により検知率が上昇する検体の割合を調べたところ，各 Group で 17%から 18.3%の検体が該当した．通常，検体から得られる情報は欠損が起きることによって低下してしまう．これは欠損が起きていること自体を悪性判定の手がかりとしている AV が存在していることを示唆している．

3.1 節において著名な AV でマルウェア名を付与できた検体群に対して，マルウェア名の種類ごとに欠損の影響を算出した結果を**表 5** に示す．名称不明の 546 検体についての結果はこの表に含まれていない．検知率が低下した種類数とは，所属する検体のうち過半数で検知率の低下が起こったマルウェア名の種類数を表し，検知率が上昇した種類数とは，所属する検体のうち過半数で検知率の上昇が起こったマルウェア名の種類数を表す．どの欠損位置においても最低 86%のマルウェア種で検知率の低下が起きており，特に Group 1 ではすべてのマルウェア種において検知率が低下している．

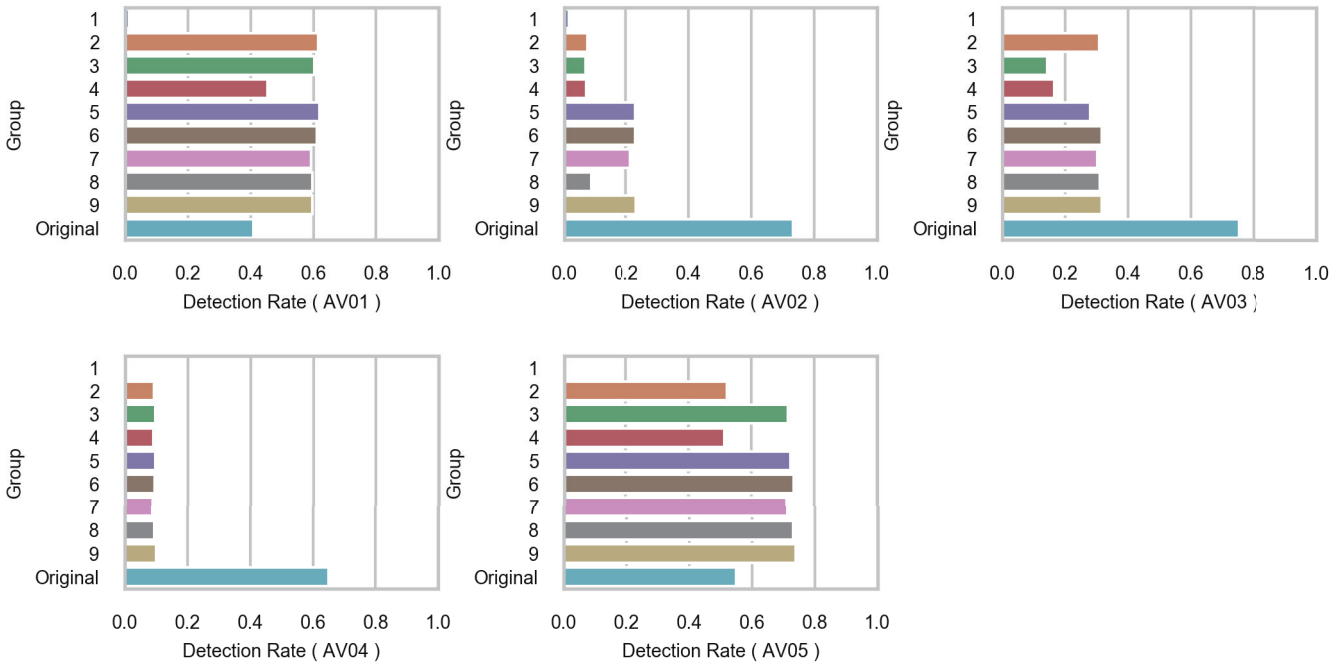

図 3 AV 別の欠損マルウェア検知率

Fig. 3 Detection rate of corrupted malware for each antivirus.

表 5 欠損により検知率が変化したマルウェア名の種類数

Table 5 The number of malware names that detection rate changed due to data loss.

	検知率が低下した種類数	検知率が上昇した種類数
Group 1	70 (100%)	0 (0%)
Group 2	64 (92%)	6 (8%)
Group 3	60 (86%)	10 (14%)
Group 4	61 (87%)	9 (13%)
Group 5	60 (86%)	10 (14%)
Group 6	60 (86%)	10 (14%)
Group 7	62 (89%)	8 (11%)
Group 8	62 (89%)	8 (11%)
Group 9	63 (90%)	7 (10%)

4.1.2 各 AV ごとの検知率

図 3 に，各 AV の欠損マルウェアに対する検知率（Detection Rate）を AV ごとに示した．4.1.1 項と異なり，本項での検知率とは，ある特定の AV が欠損マルウェアグループ全体をスキャンし，そのうち悪性と判定できた検体の割合を表す値である．スキャンに使われた AV のうち著名な物を 5 種類選出し，それぞれ AV01-05 と名付けた．

AV によらず，Group 1 に対する検知率はほぼ 0 である．これは今回選出しなかった他の AV でも同一であった．AV02-04 は，欠損前の検体に対しては AV01 や AV05 と比べて 0.1 から 0.34 程度良好な検知率を示すが，欠損により検知率が 0.4 以上低下する．逆に AV01 および AV05 は，欠損前よりも欠損後のほうが検知率が向上している．

図 3 では，欠損の有無にかかわらず検知できていなかった検体も含めて検知率を測定していたため，欠損マルウェア全体の検知率を知るためには役に立つが，欠損による純粋な影響が見えにくい．そこで，欠損による検知率への影響をより顕著に表すために，欠損させる前は各 AV で検知できていた検体だけを対象として検知率を測定した．**図 4** にその結果を示す．欠損前検体群である Original の検知率はつねに 1 である．N はその AV で欠損前の検知率が 1 であった検体の数である．

この場合においても，Group 1 に対する検知率は AV によらずほぼ 0 であり，先頭部分の欠損による検知率への影響が真に大きいことが分かる．Group 1 以外でも欠損による検知率の低下は起きており，多いもので 91%（AV02 における Group4）低下する．

4.2 欠損がマルウェア同定に与える影響

次に，マルウェアの欠損がマルウェア同定に与える影響について示す．実験結果から，欠損後の検体から欠損前のマルウェア名を導き出せているかどうかを調査した．

図 5 に，各 AV の欠損マルウェアに対する同定成功率（Accuracy Rate）を示す．同定成功率とは，各 AV を用いて欠損マルウェアグループ中の検体に対してマルウェア名を付けたとき，欠損前と同じマルウェア名を命名できた割合を示す値である．欠損前の時点で悪性と判定できなかったために欠損前のマルウェア名を付けられなかった検体については，その AV の同定成功率の計算から除外している．つまり同定成功率は，欠損前後でマルウェア名が変わらなかった検体数を，欠損前にマルウェア名を付けられた検体数で割った値である．

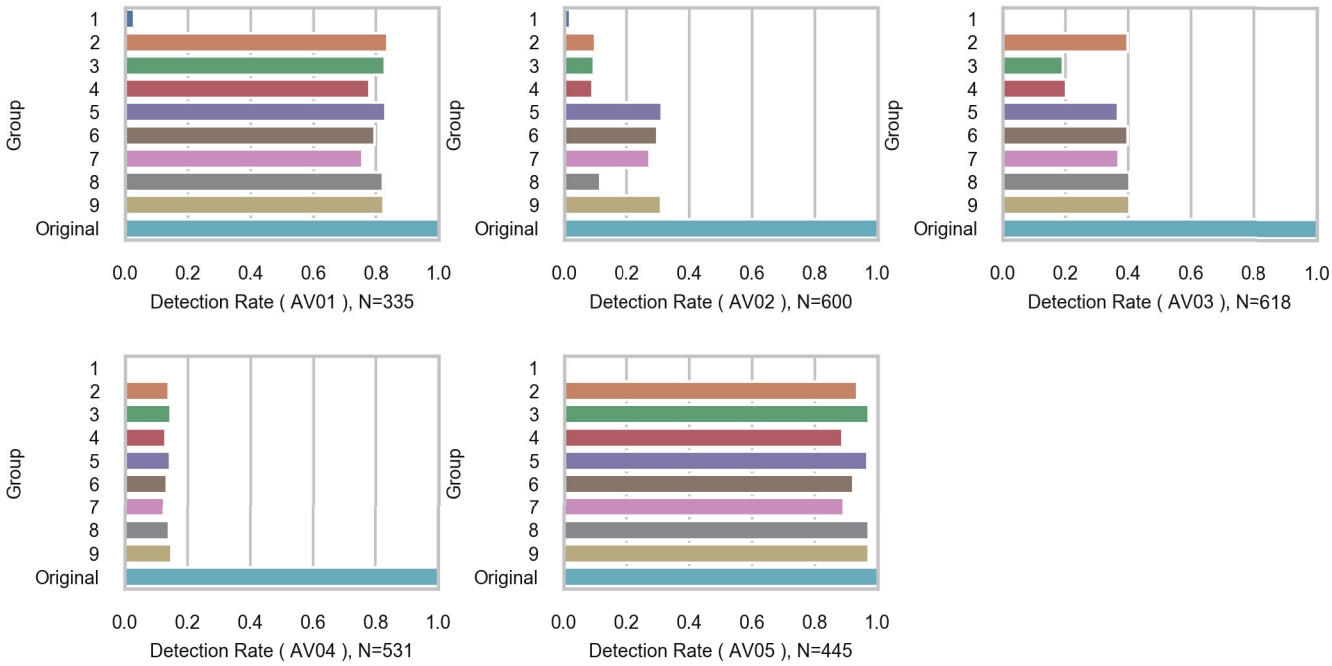

図 4 AV 別の欠損マルウェア検知率（対象とする検体を限定した場合）

Fig. 4 Detection rate of corrupted malware for each antivirus (Limited to particular malwares).

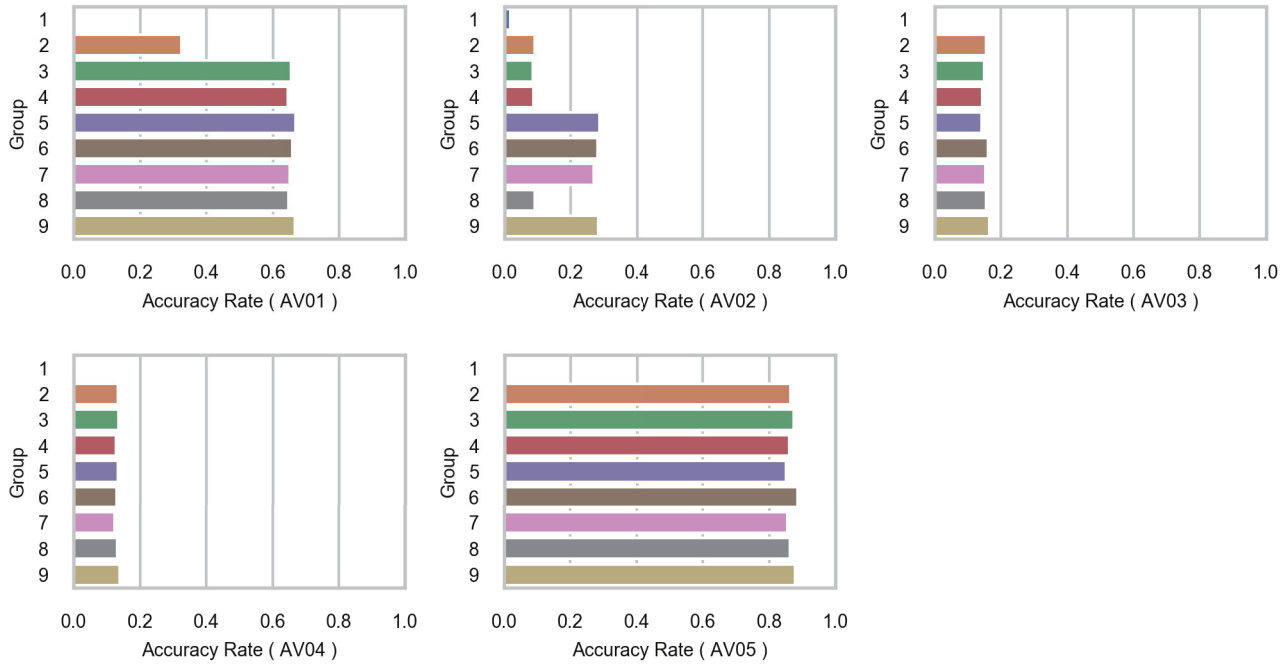

図 5 AV 別のマルウェア同定成功率

Fig. 5 Accuracy of corrupted malware for each antivirus.

同定失敗のパターンは様々である．たとえば，無欠損時にある AV に Ransom:Win64/Satwancrypt!rfn と判定されたある検体は，欠損後はその欠損位置によって Program:Win32/Unwaders.A!ml，Trojan:Win32/Fuery.B!cl，Trojan:Win32/Wacatac.B!ml，Trojan:Win32/Fuerboos.C!cl と様々な判定がなされた．無欠損時に Backdoor:MSIL/Bladabindi と判定されたある検体は，欠損後は多くの欠損位置において Trojan:Win32/Fuery.C!cl と判定された．

AV によらず，Group 1 に対する同定成功率は検知率と同様にほぼ 0 であった．AV01 は欠損後の検体に対しても比較的高い同定成功率となっており，Group 1 および Group 2 以外で 0.6 以上を維持している．AV01 の Group 2 については Group 3 以降の半分の同定成功率となっており，ファイルオフセット 0x1000 からの欠損が同定成功率の低下を引き起こしていることが分かる．AV02 は全体的に同定成

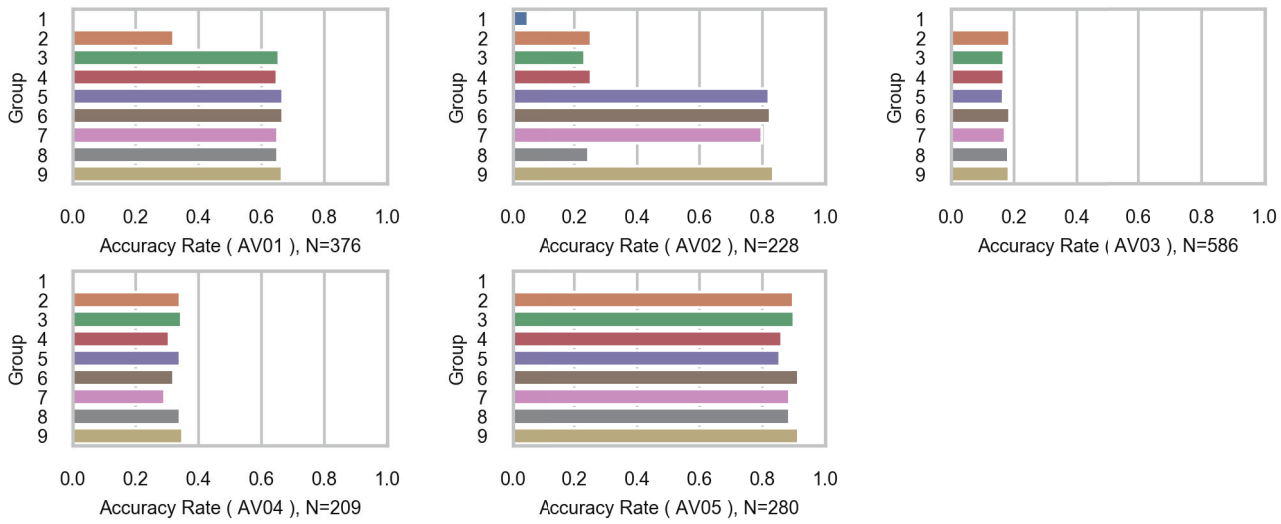

図 6 AV 別のマルウェア同定成功率（Generic 名を除外）
Fig. 6 Accuracy of corrupted malware for each antivirus (Excluding of generic name).

表 6 各 AV の Generic 名付与状況
Table 6 Naming of malware by each antiviruses.

AV	命名総数	Generic 名数	Generic 名付与率
AV01	376	0	0
AV02	661	433	0.655
AV03	684	98	0.143
AV04	569	360	0.633
AV05	501	221	0.441

表 7 欠損した良性ソフトウェアの AV による判定結果（良性ソフトウェア総数：835 件）
Table 7 Classification of corrupted benign software by antiviruses.

欠損開始 ファイルオフセット	1 つ以上の AV に 悪性と判定されたファイル数
欠損なし	8
0x0000	0
0x1000	262

功率が低いが，Group 5–7，9 は他の Group の 2 倍以上である 0.3 近くまで高くなっている．AV03 および AV04 は，欠損箇所によらず同定成功率が 0.2 未満と低くなっている．AV05 は，Group 1 以外で 0.8 以上と高い同定成功率を示している．しかし，AV05 により付けられた名称を確認すると，欠損前後ともに特定のマルウェアを表す名称ではなく Generic 名が多く付けられていた．

欠損前の検体に対して，各 AV 検知に成功し名前を付けた数と，その内で Generic 名が付けられた数を**表 6** に示す．ここで述べる Generic 名とは，ファミリ名を特定できていない名称を指す．ファミリ名まで特定できていれば，亜種名が特定できていなくても Generic 名とは見なさない．AV05 だけでなく，AV02 および AV04 も高い割合で Generic 名を検体に付与している．

欠損前の検体に非 Generic 名を付与できた検体だけを対象として，再度 AV 別のマルウェア同定成功率を算出した．その結果を**図 6** に示す．Generic 名の付与率が低かった AV01 および AV03 は対象検体を変更した影響をほとんど受けておらず，非 Generic 名に絞った場合も欠損の影響は Generic 名を除外しない場合（図 5）と同傾向であった．AV05 は Generic 名の付与率が高かったものの，Generic 名を除外しても検知率の水準はほぼ変わっておらず，高い同定成功率を示している．Generic 名の付与率が特に高い AV02 および AV04 は，図 5 からグラフの形状を保ったまま，同定成功率を大きく上げている．無欠損でもファミリ名の特定に至らなかった命名が難しい検体が除外された結果と考える．全 AV に共通して，Generic 名を除外した場合もファイル先頭からの欠損が同定成功率に最も悪影響を与えている．

これらから，マルウェアのデータ欠損は場所によらずマルウェアの同定成功率を低下させ，特にファイル先頭からの欠損は検知率の場合と同様に著しい同定成功率の低下を引き起こすといえる．ファイルオフセット 0x1000 以降の欠損は，同定に使用する AV によって影響度合いが異なるが，おおむね若いファイルオフセットから始まる欠損がより同定成功率の低下を引き起こす．

4.3 欠損した良性ソフトウェアの誤検知状況

本節では 3.2.2 項の実験 B の結果を示す．

表 7 に，欠損処理を施した良性ソフトウェアがどれだけ悪性と判定されたかを示す．どの欠損開始ファイルオフセットにおいても欠損量は 0x1000 bytes である．欠損処理を行わない場合でも，8 件の良性ソフトウェアが 1 種以上の AV に悪性と判定された．ファイル先頭からの欠損が起きた場合は，マルウェアのファイル先頭を欠損させた場合

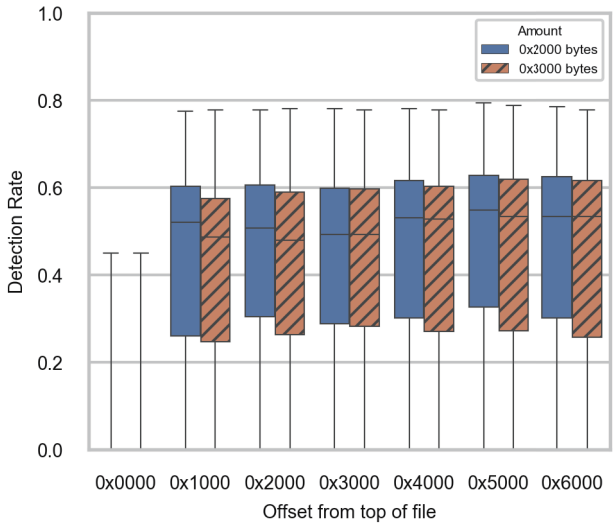

図 7 先頭付近から複数クラスタ欠損時の検知率
Fig. 7 Detection rate when multiple clusters near the top of file are lost.

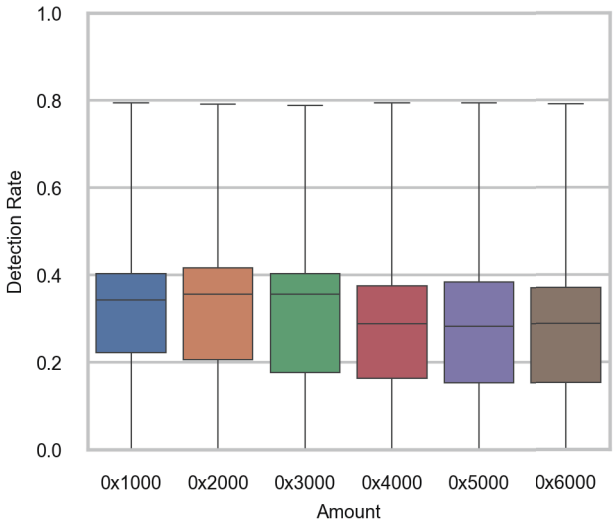

図 8 末尾から複数ブロック欠損時の検知率
Fig. 8 Detection rate when multiple blocks at the end of file are lost.

と同様に検知率が落ち込み，どの良性ソフトウェアも悪性判定されていない．ファイルオフセット 0x1000 から欠損させた場合は，262 件もの良性ソフトウェアが 1 種以上の AV に悪性と判定された．4.1 節においても言及したとおり，欠損が起きていること自体を悪性判定の手がかりとしている AV が存在していることを強く示唆している．ただし，欠損処理を施した良性ソフトウェアを悪性と判定した AV の種類数は，1 つの良性ソフトウェアに対して最大で 3 種，最頻は 1 種であった．

4.4 複数クラスタの欠損やファイル末尾からの欠損による影響

本節では 3.2.3 項の実験 C の結果を示す．

4.4.1 欠損が検知率に与える影響

本項では 4.1 節と同様に，欠損が検知率に与える影響を示す．

図 7 に，ファイル先頭付近から複数クラスタを欠損させた場合の検知率を示した．縦軸は検知率，横軸は欠損開始ファイルオフセットであり，箱部分に模様が存在しない方が 2 クラスタ分の欠損を，模様が存在する方が 3 クラスタ分の欠損を表している．1 クラスタ分の場合と同様に，ファイル先頭からの欠損は検知率に致命的な悪影響を与えている，ファイル先頭から 3 クラスタ分の欠損が起こった場合，検知率の中央値は 0，平均値は 0.011，標準偏差 0.048 と非常に低い値となっている．オフセット 0x1000 以降の欠損については，どの欠損位置でも似た傾向を示しており，欠損量が増えると検知率が微減するという自明な結果となった．

図 8 に，ファイル末尾からファイル先頭へ向かって複数ブロックを欠損させた場合の検知率を示した．縦軸は検知率，横軸は欠損量を表している．欠損量が増えるほど検知率

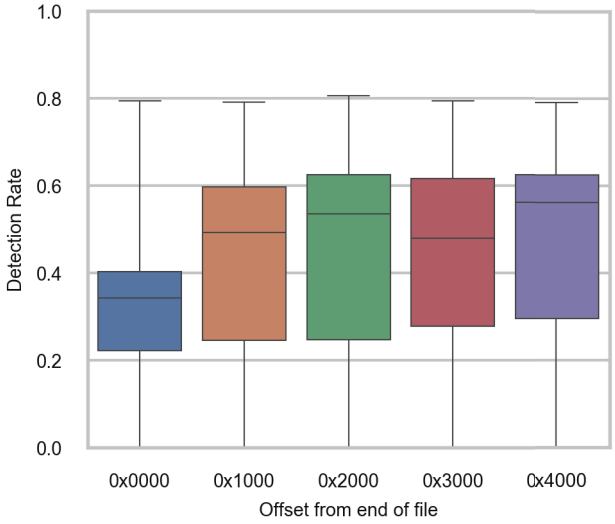

図 9 末尾付近から単一ブロック欠損時の検知率
Fig. 9 Detection rate when a block at the end of file are lost.

が低下する傾向にある．また，1 ブロック（0x1000 bytes）欠損しただけでも，図 7 においてファイルオフセット 0x1000 から 3 クラスタ欠損させた場合よりも検知率が明らかに低い．

図 9 に，ファイル末尾付近から単一ブロックを欠損させた場合の検知率を示した．縦軸は検知率，横軸はファイル末尾から先頭方向への欠損開始ファイルオフセットを表している．たとえば横軸の値が 0x1000 のグラフは，ファイル末尾からファイル先頭方向へ 0x1000 bytes さかのぼった位置からファイル先頭へ向かって 1 ブロック分欠損させたときの検知率を表している．ファイル末尾（0x0000）から 1 ブロックの欠損が最も悪影響を与えており，ファイル先頭からの欠損に次いで検知率が低くなっている．ファイル末尾から 0x1000 bytes 以上さかのぼった位置から欠損させた場合の検知率は，ファイル先頭からのオフセット 0x1000

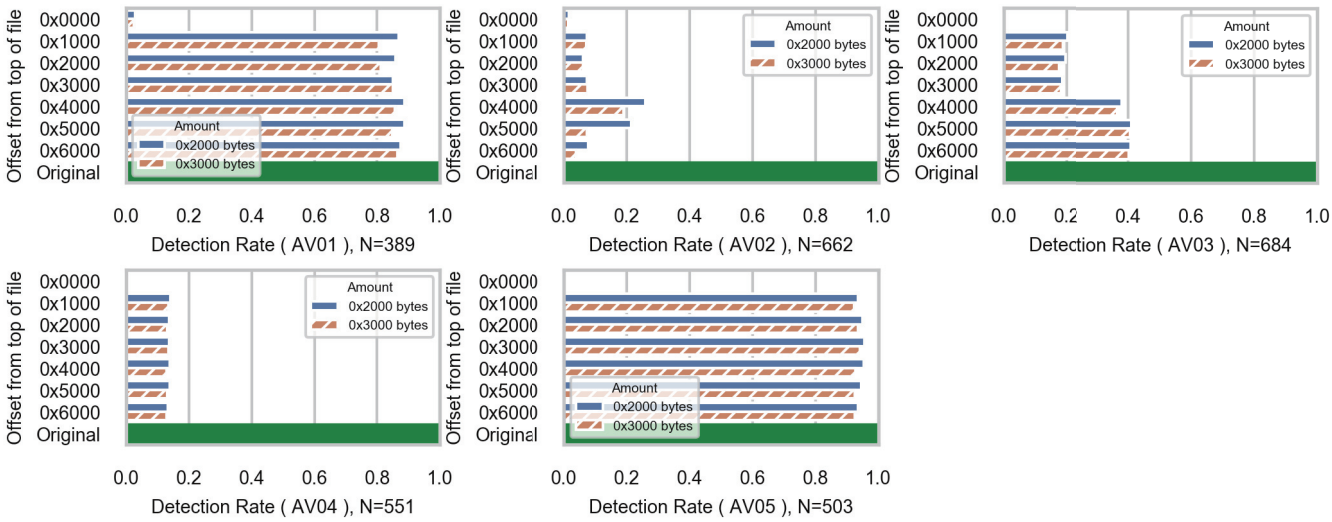

図 10 先頭付近から複数クラスタ欠損時の AV 別の欠損マルウェア検知率

Fig. 10 Detection rate for each antivirus when multiple clusters near the top of file are lost.

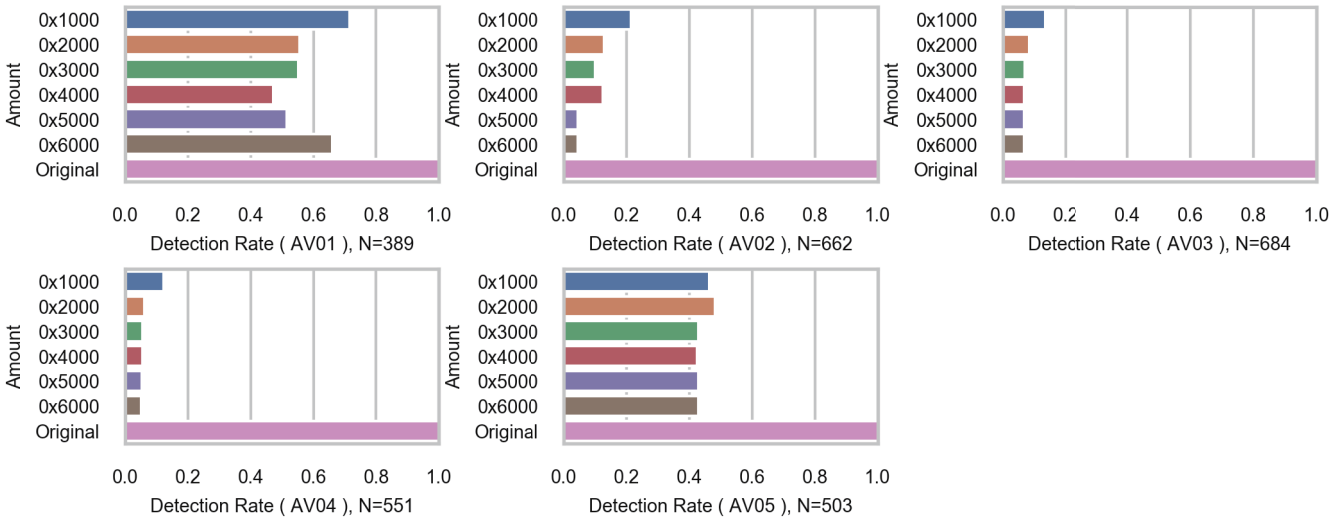

図 11 末尾から複数ブロック欠損時の AV 別の欠損マルウェア検知率

Fig. 11 Detection rate for each antivirus when multiple blocks at the end of file are lost.

から始まる欠損と大差がない．このことから，前述の図 8 の結果においても，ファイル末尾からの 1 ブロック分の欠損が検知率に最も大きく影響を与えていたと推測できる．

4.4.2 各 AV ごとの検知率

本項では 4.1.2 項と同様に，各 AV の欠損マルウェアに対する検知率を示す．検知率の算出対象とする検体も 4.1.2 項と同様に，無欠損時は対象の AV で検知ができていた検体に限っている．

図 10 に，ファイル先頭付近から複数クラスタを欠損させた場合の検知率を AV 別に示した．縦軸は欠損開始ファイルオフセット，横軸は検知率であり，棒部分に模様が存在しない方が 2 クラスタ分の欠損を，模様が存在する方が 3 クラスタ分の欠損を表している．各グラフ最下部の Original は無欠損検体を表しており，その検知率は本項ではつねに 1 である．単一クラスタのみを欠損させた場合とグラフ形状にほぼ差はない．どの欠損開始ファイルオフセットにおいても，欠損させたクラスタ群の中で最も検知率に悪影響があった特定クラスタに影響を受けた検知率となっている．

図 11 に，ファイル末尾からファイル先頭へ向かって複数ブロックを欠損させた場合の検知率を AV 別に示した．縦軸は欠損量であり，どれもファイル末尾からの欠損である．どの AV においても，末尾から 1 ブロック分欠損させた時点で検知率が大きく落ちており，それ以降は欠損量と比例して検知率が低下する傾向が見て取れる．末尾 1 ブロックにも検知の重要な手がかりが存在していることがうかがえる．

図 12 に，ファイル末尾付近から単一ブロックを欠損さ

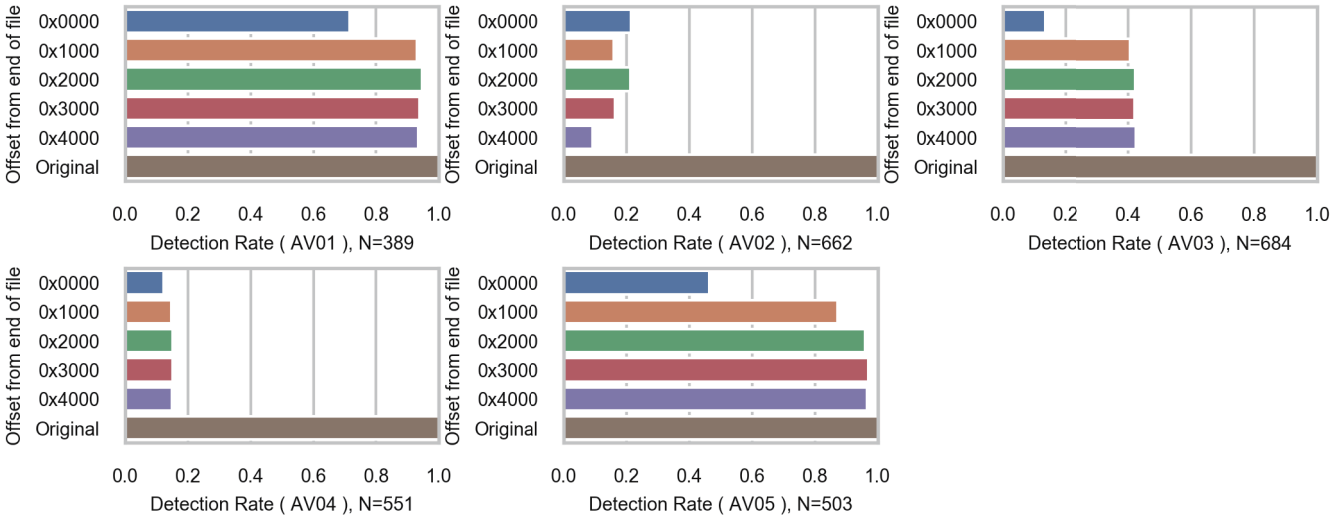

図 12 末尾付近から単一ブロック欠損時の AV 別の欠損マルウェア検知率

Fig. 12 Detection rate for each antivirus when a block at the end of file are lost.

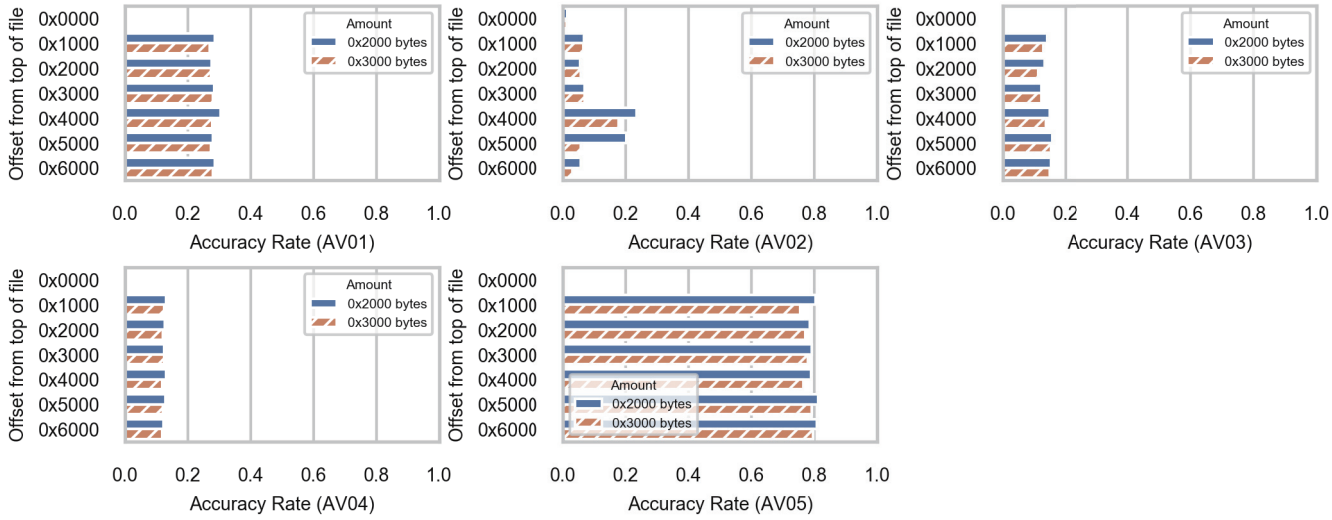

図 13 先頭付近から複数クラスタ欠損時の AV 別のマルウェア同定成功率

Fig. 13 Accuracy for each antivirus when multiple clusters near the top of file are lost.

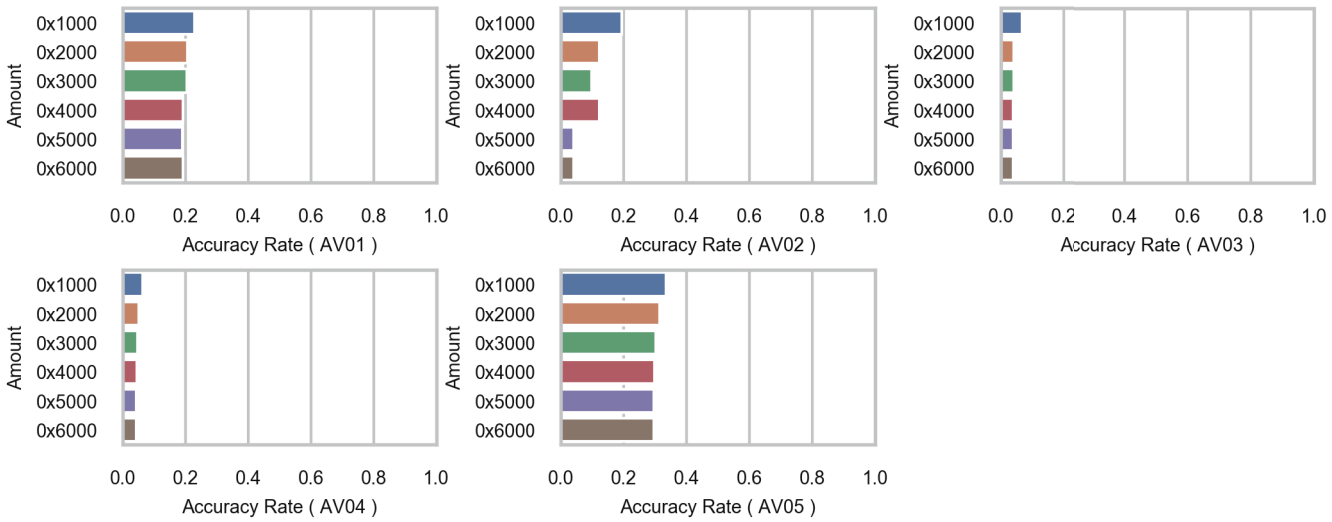

図 14 末尾から複数ブロック欠損時の AV 別のマルウェア同定成功率

Fig. 14 Accuracy for each antivirus when multiple blocks at the end of file are lost.

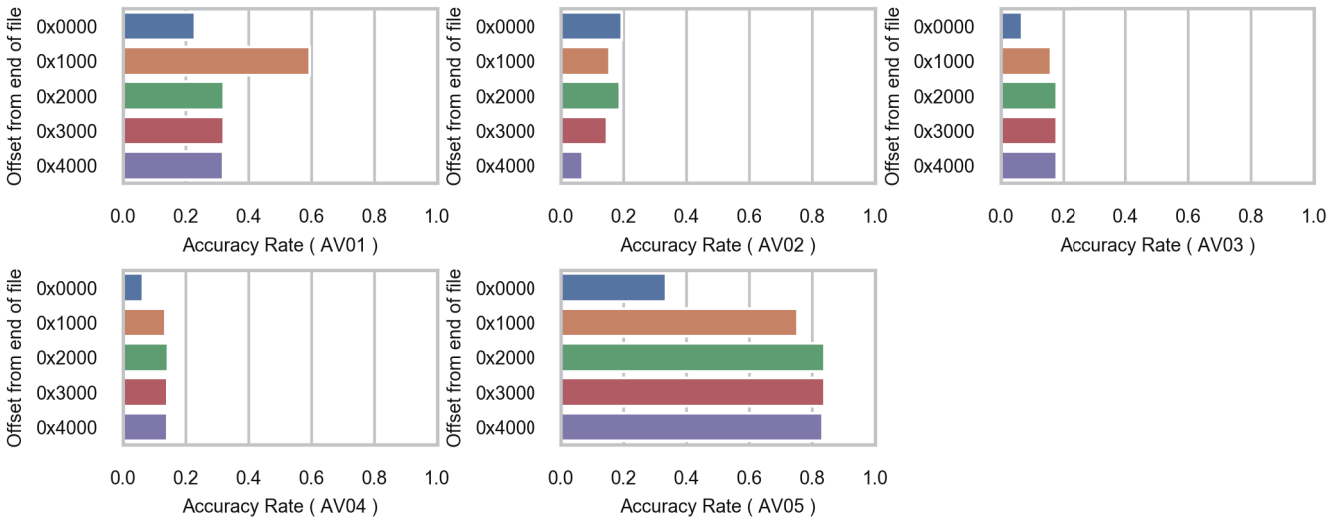

図 15 末尾付近から単一ブロック欠損時の AV 別のマルウェア同定成功率
Fig. 15 Accuracy for each antivirus when a block at the end of file are lost.

せた場合の検知率を AV 別に示した．縦軸はファイル末尾から先頭方向への欠損開始ファイルオフセットを表しており，欠損量はつねに 1 ブロックである．どの欠損位置においても検知率は Original より低下しており，AV02 を除く著名 AV で，末尾 1 ブロックを欠損させた場合の検知率の低下が，他の末尾付近ブロックを欠損させた場合よりも大きい．

4.4.3 欠損がマルウェア同定に与える影響

本項では 4.2 節と同様に，マルウェアの欠損がマルウェア同定に与える影響について示す

図 13 にファイル先頭付近から複数クラスタを欠損させた場合の同定成功率を AV 別に示した．ファイル先頭からの欠損が同定成功率の致命的な低下を引き起こすことはこれまで述べてきたとおりである．AV01 のオフセット 0x1000 および AV02 から AV05 は，単一クラスタを欠損させた場合と同様の傾向を示しており，AV01 のオフセット 0x2000 以降は，単一クラスタを欠損させた場合よりも同定成功率が大きく落ち込んでいる．

図 14 にファイル末尾からファイル先頭へ向かって複数ブロックを欠損させた場合の同定成功率を AV 別に示した．検知率と同様に，どの AV においても末尾から 1 ブロック分欠損させた時点で同定成功率が大きく落ちており，それ以降は欠損量と比例して同定成功率が低下する傾向が見て取れる．AV02 を除いて，末尾 1 ブロックを欠損させた時点で，Group 2 以降のどの単一クラスタの欠損よりも同定成功率が低くなっている．このことから，末尾の情報はファイル先頭部分（Group 1）に次いで同定に重要な手がかりとなっていることが見て取れる．

図 15 にファイル末尾付近から単一ブロックを欠損させた場合の同定成功率を AV 別に示した．この場合も AV02 を除いて末尾 1 ブロックを欠損させた場合の同定成功率が最も低く，多くの著名 AV で末尾 1 ブロックの欠損が同定に強い悪影響を与えることが分かる．

5. 考察

実験 A の結果である 4.1 節および 4.2 節から，若いファイルオフセットからの欠損，特にデータ先頭からの欠損が検知率およびマルウェア同定の成功率を低下させることが分かった．そこで，各欠損マルウェアグループで具体的にどのような意味のデータが欠損しているかを，検体の PE ファイルとしての構造 [3] に着目して解析し調査した．

Group 1 の欠損箇所であるファイルオフセット 0x0000 から 0x1000 bytes 分の範囲には，IMAGE_DOS_HEADER 構造体などからなる MS-DOS スタブ，IMAGE_NT_HEADERS 構造体に含まれるシグネチャや，IMAGE_FILE_HEADER 構造体からなる COFF ファイルヘッダ，データディレクトリを含む IMAGE_OPTIONAL_HEADER 構造体からなるオプショナルヘッダといった，PE ファイルにおいて非常に重要な情報や，IMAGE_SECTION_HEADER 構造体から構成されているセクションテーブルが格納されている．今回収集した検体群においては，これら情報は Group 2 以降の欠損箇所であるファイルオフセット 0x1000 より後には含まれていなかった．また，IMAGE_DIRECTORY_ENTRY_IAT 用の IMAGE_DATA_DIRECTORY 構造体が指しているインポートアドレステーブルや IMAGE_DIRECTORY_ENTRY_DEBUG 用の同構造体が指しているデバッグディレクトリが格納されている領域が，ファイルオフセット 0x1000 以降に比べて多く含まれていた．IMAGE_DIRECTORY_ENTRY_IMPORT 用の IMAGE_DATA_DIRECTORY 構造体が指しているインポートテーブルが格納されている領域も含まれていたが，他の Group と比較して多く含まれているわけではなかった．このように，ファイルオフセット 0x0000 から 0x1000 bytes 分の範囲には，検体を PE ファイルと断定す

るためのデータや検体中の重要なデータへのポインタといったヘッダ情報の多くが格納されているため，この領域の欠損は AV によるシグネチャマッチングやバイナリ解析に影響を与え，検知率や同定成功率を著しく下げていると考えられる．

セクションテーブルが指しているセクションデータ本体は，どのセクションもファイル先頭付近の各欠損箇所にはほぼ同量出現しており，特色は見られなかった．コードが格納されている箇所も，特定の位置に集中していなかったため，このような情報を基にマルウェア検知や同定に使う手法は先頭付近が欠損したマルウェアに対して有効であると思われる．

Group 2（ファイルオフセット 0x1000）および Group 4（ファイルオフセット 0x3000）は，ファイル先頭からの欠損においては Group 1 に次いで検知率の中央値が低かった．Group 2 で欠損した領域には，Group 1 ほど顕著ではないものの IMAGE_DIRECTORY_ENTRY_IAT から参照されるインポートアドレステーブル関連の情報が Group 3 以降よりも多く含まれていた．Group 4 で欠損した領域に関しては，Group 3 および 5 以降で欠損した領域と比較しても，今回の調査においてはあまり差異が見られなかった．

実験 B の結果である 4.3 節から，一部の AV は良性ファイルであっても欠損していることにより悪性と判定する場合があると分かった．ただしそのような AV は少なく，誤判定はある特定の 3 種の AV が最も多く起こしていた．これら 3 種の AV は 4 章の著名 AV には入っていない．

実験 C の結果である 4.4 節から，ファイル末尾付近の欠損，特にファイル末尾から 1 ブロック（0x1000 bytes）の欠損が検知率およびマルウェア同定の成功率を低下させることが分かった．検体を解析したところ，これらの欠損位置には.rsrc セクションが多く格納されていた．この.rsrc セクションは通常はリソース情報を格納しており，リソース情報は検体を特徴付ける重要な情報であるため，このブロックの欠損が大きく悪影響を与えていると考えられる．

6. 関連研究

Shafiq ら [4] は，機械学習アルゴリズムを使用して PE ファイルフォーマットの構造的特徴からファイルの良性・悪性判定をリアルタイムで判定する仕組みである PE-Miner を提案しており，テスト環境で 99%を上回る検知率と 0.5%を下回る誤検知率を出している．ファイル先頭付近の構造から得られる情報を有用な特徴量として採用しているため，Group 1 のようなファイル先頭からの欠損はこういった機械学習手法による悪性検知に対しても致命的な検知率の低下を引き起こすものと考えられる．

Hand ら [1] は，マルウェアが自身を削除したときなど x86 用の実行可能ファイルが削除されたとき，それを復元する手法 Bin-Carver を提案している．EXT2 ファイルシステムと ELF 形式の実行可能ファイルにおいて，削除されたファイル群のうち 93.1%のファイルを復元することに成功している．ただし，ELF フッタやファイル先頭に存在する ELF ヘッダを復元のための重要な情報として扱っているため，ファイル先頭が欠損した検体の復元にどれほど適用できるかは未知数である．

大坪ら [5] は，コードの断片からコンパイラを推定する手法について提案している．16 命令分の小さなコード断片からコンパイラの種類を推定しており，コード断片の位置によらず高い精度を出している．マルウェアの種類を直接的に推定するわけではないものの，欠損マルウェアのコンパイラ情報が分かれば，あるマルウェアと同一のコンパイラが使われているといった情報が得られるため，欠損位置にかかわらずに使用できるマルウェア同定のための指標として使用できる可能性がある．

横瀬ら [6] は，word2visualvec の学習モデルをベースに，malware2mutualvec というマルウェアの亜種検知を行う学習モデルを提案している．マルウェア検体中の文字列情報および API コール列情報という 2 つの特徴を 1 つの特徴空間に射影し学習することで，表層解析や静的解析により得られる文字列情報をもとに，その検体の API コール列相当の情報を推測している．欠損マルウェアは多くの場合欠損前のように動作しないため動的解析により API コール列を得ることは難しいが，一部の欠損であれば文字列情報は残存すると考えられるため，このような手法により間接的に API コール列を推測しその情報をもとにマルウェア同定を行うことができる可能性を示唆している．また，表層解析結果や静的解析から API コール列を推測することができれば，欠損マルウェアに対しても動的解析結果を基にしたマルウェア検知・同定手法が適用できると考えられるため，利点は大きい．

動的解析結果を基にしたマルウェア同定手法としては，堀合ら [7] の研究があげられる．動的解析の結果を利用して，マルウェア間の挙動の類似性からマルウェアの名称を自動的に推定する手法を提案しており，80%以上の一致率を出している．ただし，API コール列から得られる挙動情報以外に AV によるスキャン結果も特徴として使用しているため，その特徴に関しては静的解析情報から推定した API コール列から得ることはできない．

可視化手法によりマルウェアの検知や同定を行う研究も存在している．Yoo [8] は良性の実行可能ファイルに寄生する Windows 用マルウェアを，自己組織化マップ（SOM）を使用して可視化する方法を提案している．同系統のマルウェアに感染した実行可能ファイルは類似した見た目をもつ SOM として可視化されるため，部分的な欠損の影響が少ないマルウェア同定手法として使用できる可能性がある．伊沢ら [9] は，IoT マルウェア（非 PE 形式，非 x86-64 アーキテクチャ）の逆アセンブルコードを基に類似度行列を作

成し，t 分布型確率的近傍埋め込み法で二次元平面上に可視化している．未知の検体が既知の検体の近傍にマップされており，未知検体のファミリ名特定に成功している．ただし，Windows マルウェア（PE 形式を含む）ではパッキングが施されたマルウェアも多いため，バイナリから得られる情報を解析に有効活用ができないとも述べられている．

菅野ら [10] は，doc2vec および SVM を使って，Windows 用実行ファイル内の可読文字列情報から未知のマルウェアを検知する手法を提案しており，Accuracy 0.97 を出している．マルウェア同定については言及されていないものの，マルウェア検知においては欠損マルウェアに対してもある程度耐性のある検知手法であると期待できる．

欠損マルウェアを検知や同定の対象として定めた場合，欠損位置に存在する特徴量は学習に使用できないため，特定の箇所に集中して出現していない特徴量を用いる機械学習手法が特に有効であると考える．

7. まとめと今後の課題

本論文では，ファイルを消去されたことによりデータ欠損を起こしたマルウェアに着目し，欠損がマルウェア検知率やマルウェア名の同定に与える影響について明らかにした．ファイル先頭付近の欠損においてはファイルデータの先頭から始まる 0x1000（4,096）bytes 分の欠損が最も検知率への影響が大きくほぼ 0 まで検知率が低下する．それに次いで，ファイルオフセット 0x1000 および 0x3000 から始まる 0x1000 bytes 分の欠損による検知率の低下が大きかった．欠損位置によらずマルウェア名の同定成功率は 7 割前後低下するが，ファイルオフセット 0x2000 以降から始まる 0x1000 bytes の欠損であればアンチウイルス製品（AV）によっては 6 割強の同定成功率を維持できた．ファイル末尾付近からの欠損においては，ファイルデータの末尾から先頭方向へ 0x1000 bytes 分の欠損が最も検知率の低下につながった．ファイル先頭からの欠損ほど致命的な検知率の低下はないが，多くの著名 AV でファイル先頭からのオフセット 0x1000 からの欠損よりも検知率，同定成功率ともに下回っている．また，良性ソフトウェアを欠損させることで一部の AV が悪性と判定してしまう事象についても観測した．

今後の研究課題としては，今回は x86-64 用のマルウェアのみを対象としたため x86 用のマルウェアも対象としデータの種類や数を増やして再度実験することや，欠損位置を増やすこと，ファイルオフセット 0x3000 からの欠損による影響が大きい理由を調査することがあげられる．

また，クラスタサイズを意識せずに PE 形式のファイルにおける重要な情報が格納されているヘッダのみ消去するなど，意味的なまとまりで欠損を起こした場合に検知率や同定率にどのような影響が起こるかも興味深い．

参考文献

[1] Hand, S., Lin, Z., Gu, G. and Thuraisingham, B.: BinCarver: Automatic Recovery of Binary Executable Files, *Proc. 12th Annual Digital Forensics Research Conference* (*DFRWS'12*) (2012).

[2] 林　健，佐々木良一：時間経過に着目した HDD のデータ復元に関する実験と解析，情報処理学会研究報告，Vol.2013-CSEC-60, No.14 (2013).

[3] Microsoft: Windows Dev Center - PE Format, available from ⟨https://docs.microsoft.com/en-us/windows/win32/debug/pe-format⟩.

[4] Shafiq, M.Z., Tabish, S.M., Mirza, F. and Farooq, M.: PE-Miner: Mining Structural Information to Detect Malicious Executables in Realtime, *Recent Advances in Intrusion Detection*, pp.121–141 (2009).

[5] 大坪雄平, 大塚　玲, 三村　守, 榊　剛史, 受川　弘, 岩田吉弘：コード断片からのコンパイラ推定手法，コンピュータセキュリティシンポジウム 2018 論文集，pp.1259–1265 (2018).

[6] 横瀬謙信, 大久保潤, 三村　守, 榊　剛史, 受川　弘, 岩田吉弘：word2visualvec を応用したマルウェア亜種推定の枠組みの提案，コンピュータセキュリティシンポジウム 2019 論文集，pp.1047–1051 (2019).

[7] 堀合啓一，今泉隆文，田中英彦：マルウェア亜種の動的挙動を利用した自動分類手法の提案と実装，情報処理学会論文誌，Vol.50, No.4, pp.1321–1333 (2009).

[8] Yoo, I.: Visualizing Windows Executable Viruses Using Self-organizing Maps, *Proc. 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, *VizSEC/DMSEC '04*, pp.82–89 (online), DOI: 10.1145/1029208.1029222 (2004).

[9] 伊沢亮一, 班　涛, 鉄　穎, 吉岡克成, 井上大介：IoT マルウェアに対する静的解析に基づいた類似度の有効性検証，情報処理学会研究報告，Vol.2018-SPT-27, No.19 (2018).

[10] 菅野賢輝，三浦紘弥，三村　守，田中秀磨：Doc2Vec を用いた静的解析によるマルウェア検知，情報処理学会第 80 回全国大会講演論文集，Vol.2018, No.1, pp.547–548 (2018).

小久保 博崇

2012 年筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻修了．同年より株式会社富士通研究所所属．サイバーセキュリティに関する研究に従事．

大山 恵弘　（正会員）

2001 年東京大学大学院理学系研究科情報科学専攻修了．科学技術振興事業団研究員，東京大学大学院情報工学系研究科助手，電気通信大学大学院情報理工学研究科准教授を経て，2016 年より筑波大学システム情報系准教授．博士（理学）．システムソフトウェア，ソフトウェアセキュリティに関する研究に従事．