# 標的型攻撃情報共有のための文書型マルウェアの墨塗り手法

齊藤　真吾† 吉岡　克成† 神薗　雅紀‡* 星澤　裕二‡ 松本　勉†
†横浜国立大学 大学院 環境情報学府・環境情報研究院
240-8501 神奈川県横浜市常盤台 79-7
saito-shingo-tk@ynu.jp, {yoshioka, tsutomu}.ynu.ac.jp
‡株式会社セキュアブレイン
102-0083 東京都千代田区麹町 2-6-7 麹町 RK ビル 4F
*独立行政法人 情報通信研究機構 184-8795 東京都小金井市貫井北町 4-2-1
{masaki_kamizono, yuji_hoshizawa}@securebrain.co.jp

**あらまし** 近年，企業や政府機関を狙って機密情報を漏洩させる標的型攻撃による被害が増加している．リスクの把握及びリスクを共有するためには検体を提供する必要があるが，標的型攻撃はその攻撃の入り口としてメールを利用したものが多く，標的となった個人・組織に関連する情報を含む添付文書ファイルを用いるため，検体をそのまま外部に提供しづらいという問題がある．そこで検体共有促進のために，検体の挙動に影響を与えることなく標的となった個人・組織に関連する情報を消去(墨塗り)する技術が求められている．本研究では，標的型攻撃で利用されることが多いPDF形式の検体に関してその挙動に影響を与えるテキストやメタデータなどのコンテンツ部分を特定し，当該箇所以外を削除または固定値に置換する墨塗り手法を提案する．また，実際の検体を用いてその有効性を示す．

# Sanitizing sensitive contents from document malware for accelerating sample sharing

Shingo Saito† Katsunari Yoshioka‡ Masaki Kamizono‡*
Yuji Hoshizawa‡ Tsutomu Matsumoto‡
†Yokohama National University Graduate School of Environment and Information Sciences Faculty of Environment and Information Sciences.
79-7 Tokiwadai, Hodogaya-ku, Yokohama, Kanagawa 240-8501, JAPAN
saito-shingo-tk@ynu.jp, {yoshioka, tsutomu}@ynu.ac.jp
‡Securebrain Corporation
Kojimachi　RK Bldg, 2-6-7 Kojimachi, Chiyoda-ku, Tokyo 102-0083, JAPAN
‡National Institute of Information and Communications Technology 4-2-1
Nukui-Kitamachi, Koganei, Tokyo, 184-8795 JAPAN
{masaki_kamizono, yuji_hoshizawa}@securebrain.co.jp

**Abstract** Recently, enterprises and government agencies are suffering from information leaks caused by targeted cyber-attacks. The most common attack vector is via email with document malware. Such document malware often contains texts and metadata that can specify the target of the attack, which often discourages information sharing. In this study, we propose a sanitizing method that can erase the contents unrelated to the behavior of the PDF document malware to accelerate information sharing.

# 1　はじめに

近年，企業や政府機関を狙って機密情報を漏洩させる標的型攻撃による被害が増加しており，その手法も巧妙化している．攻撃者は，標的に関する情報を長い時間かけて収集し，その情報を基に作成した文書マルウェアを添付したメールを送り，標的に添付ファイルを開かせることで文書閲覧用のソフトウェアの脆弱性を突き，標的組織内に侵入する場合が多い．標的型攻撃メールは送信元を実在の組織に偽装したり，件名や本文，添付文書ファイル名を信頼できる内容にしたりする上，添付文書ファイルの内容もマルウェアが埋め込まれていると気付かれないように攻撃対象の個人や組織に関連する内容となっている場合が多い．

このような標的型攻撃への対策技術の研究開発を進めるためには，標的となった個人・組織が文書型マルウェア検体を研究機関等に提供することにより，攻撃に関する情報を共有することが重要である．しかし前述の通り，文書型マルウェア検体は標的となった個人・組織の情報を含んでいるため，共有時には，これらの情報を消去しなければならない場合がある．本研究では，これを文書型マルウェアの墨塗りと呼ぶこととする．文書型マルウェアの中には，感染時にテキスト，画像，メタデータなどの文書ファイル内のコンテンツを参照して動作するものが多く存在するため，これらを考慮して墨塗りを施す必要がある．

本研究では，標的型攻撃で利用されることが多い PDF 形式の文書型マルウェアを対象とし，その挙動に影響を与える恐れのあるテキストデータやメタデータの箇所を分析により特定した上で，当該箇所を避けて墨塗りの必要な箇所を削除または固定値に置換する手法を提案する．提案手法では動的解析を利用した難読化 JavaScript コード解析システム[1]を利用して得られる API フックログの情報から文書型マルウェア検体内で悪性コードに参照されているテキストやメタデータの箇所を特定する．実検体 10 件を用いた評価実験では，墨塗り対象の情報が意図通り消去されており，かつ，墨塗り後の文書型マルウェア検体と墨塗り前の文書型マルウェア検体の挙動に差異がないことをサンドボックス解析により確認する．

本稿の構成は以下の通りである．まず 2 章で文書情報を参照する PDF 形式のマルウェア検体の例を説明し，3 章で提案手法について説明する．4 章では実検体を用いた評価実験について述べ，5 章でまとめと今後の課題を述べる．

# 2　PDF 形式のマルウェア

本章では PDF 形式の文書型マルウェアについて今回の評価実験で用いた検体を例に挙げて説明する．2.1 節では PDF 形式のマルウェアの特徴について述べ，2.2 節では文書内のコンテンツを参照するための JavaScript for Acrobat API に関して説明し，2.3 節では具体的な検体の例を説明する．

## 2.1　特徴

本節では PDF 形式の文書型マルウェアが持つ特徴について説明する．Adobe Reader にはこれまで複数の脆弱性が発見されており，それらを悪用したマルウェアが多く存在する．その多くには不正な JavaScript コードが難読化されて組み込まれている．難読化の方法は多様であり，PDF 形式の文書ファイル内に EXE 形式のマルウェアを埋め込むことも可能なので，アンチウイルスソフトによって検知することが難しい．

## 2.2　文書内のコンテンツを取得する JavaScript for Acrobat API

一般に PDF 文書には JavaScript を埋め込むことが可能であるため，攻撃者は Adobe Reader の JavaScript エンジンの脆弱性を悪用して攻撃を行うことが多く，埋め込む

JavaScript はアンチウイルスソフトの検知を逃れるために難読化されることが多い．難読化の際に JavaScript for Acrobat API を使用し，文書の特定ページ内の単語や単語数，メタデータを参照することがある．

### 2.3　実検体例

本節では文書内のコンテンツを参照し，その内容に応じて動作を変える検体の一例を示す. 以下の図 1 と図 2 は D3M Datasets 2010 の PDF 形式の文書型マルウェア検体（MD5: e72f100de9bb68cb2f92b18efe02eb60）を動的解析した際の API フックログである. 図 1 では JavaScript for Acrobat API の getPageNthWord メソッドにより指定ページの指定位置の単語を取得し，その一部を繰り返し変換することで最終的に攻撃部分の関数が現れるような難読化を行っている．この例では，指定ページの単語数を取得するメソッドである getPageNumWords と併せて用いられており，指定ページ内の全ての単語が参照されている．また，図 2 ではメタデータ(Author)の情報を参照しており，悪性サイトと思われる URL を取得していることが分かる．

```
function get_url() {
    var str = this.info.author;
    var ret = encode_str(str, dest_table, src_table);
    return ret;
};
```

図 2．PDF 文書内のメタデータを参照する文書型マルウェア検体の API フックログ

## 3　墨塗り手法の提案

本章では標的型攻撃情報共有のための検体

```
      no => 0
      expression =>   var p = ["un", "ca", "es", "pe", ""];
  l = new String(p[0] + p[2] + p[1] + p[3] + p[4]);
  var google = ["", "t", "deA", "cha", "rCo"];
  v = new String(google[3] + google[4] + google[2] + google[1] + google[0]);
  var s = ["umWo", "", "getP", "rds", "ageN"];
  y = new String(s[2] + s[4] + s[0] + s[3] + s[1]);
  var yGoogle = ["eNt", "", "get", "hWo", "Pag", "rd"];
  kO = new String(yGoogle[2] + yGoogle[4] + yGoogle[0] + yGoogle[3] + yGoogle[5] + yGoogle[1]);
  var t = ["ev", "al", ""];
  e = new String(t[0] + t[1] + t[2]);
  var b = ["p", "a", "p", ""];
  d = new String(b[1] + b[0] + b[0] + b[3]);
  var x = ["arC", "mCh", "fro", "", "ode"];
  sV = new String(x[2] + x[1] + x[0] + x[4] + x[3]);
  var uV = 2;
  var a = 20;
  var d = this[d];
  var m = String("%");
  var pGoogle = new String();
  var zV = this[y](uV);
  var update = this[l];
  for (var k = 0; k < zV; k++) {
      adobe = this[kO](uV, k);
      var g = adobe.substr(adobe.length - 2, 2);
      var r = update(m + g);
      var sD = r[v](0);
      var iS = sD ^ a;
      pGoogle += String[sV](iS);
  }
  this[e](pGoogle);
[global: getPageNumWords]
      page => 2
[global: getPageNumWords]
      page => 2
      n => 0
[global: unescape]
      input => %1e
```

図 1.PDF 文書内のテキスト部分を参照する文書型マルウェア検体の API フックログ

文書ファイルの墨塗り手法を提案する．まず 3.1 節では墨塗り手法の要件を述べ， 3.2 節で墨塗りシステムの説明を行い，3.3 節で具体的な墨塗り処理を説明する．

## 3.1　要件

2 章で述べた通り，文書型マルウェア検体は，テキストや画像，メタデータといった墨塗り対象コンテンツと，脆弱性を突いて不正な処理を行う JavaScript や EXE ファイルといった悪性コードから成る．この墨塗り対象コンテンツの一部および全部を削除または固定値に置換する処理を墨塗りと呼ぶこととする．本研究では，文書型マルウェア検体を共有しその不正な動作を解析することで攻撃への対策を行うことを目的としているため，文書型マルウェア検体内の悪性コードの動作に影響を与えないことが墨塗り処理の最低限の要件である．

## 3.2　墨塗りシステム

本節では墨塗りシステムについて説明する．墨塗りシステムの概要図は図 3 のようになっている．

- PDF パーサ

システムへの入力である文書型マルウェア検体から JavaScript と墨塗り対象コンテンツ情報(テキスト，メタデータ)を取得する．

- JavaScript インタプリタ

PDF パーサが抽出した JavaScript を実行する．

- 疑似 JavaScript for Acrobat API 環境

PDF パーサが取得した PDF 文書のコンテンツ情報を参照し，JavaScript インタプリタに対して，JavaScript for Acrobat API を実装した環境を用意し，JavaScript for API が呼ばれた際には PDF 文書内で JavaScript が動作するのと同様の機能を疑似的に提供する．この際，コンテンツ情報を参照する API については呼び出し履歴をログとして記録する．

- PDF サニタイザ

文書型マルウェア検体の墨塗り対象コンテンツに対して墨塗り処理を行う．この際，墨塗り対象コンテンツを参照する API フックログを参照し，参照されている箇所を特定し，それ以外の箇所に対して墨塗りを行うことで，悪性コードの動作が墨塗りにより変化しないようにする．PDF サニタイザの処理については 3.3 節で説明する．

## 3.3　墨塗り処理

PDF サニタイザの処理を図 4，5 に示す．

まず，墨塗り対象の文書型マルウェア検体の墨塗り対象コンテンツの中でも特にテキストに関して墨塗り処理を行う．一般に PDF 文書においてページコンテンツに関する描画命令はデータ圧縮が施してあるため，文章の記述命令部分を墨塗りするためにはデータを一度展開する必要がある．そこで，PDF の操作ツールである pdftk[2]の uncompress 機能を利用し，墨塗り対象文書のデータを展開する．展開されたデータからテキスト情報を描

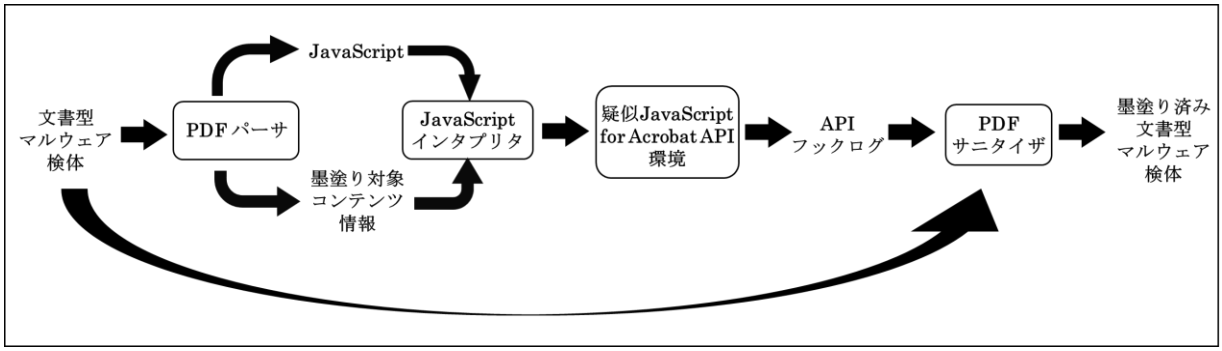


図 3.墨塗りシステム概要図

画する演算子である Tj の対象となる変数(文字列)によりテキスト情報の位置を特定し，データの書き換えを行う．但し，JavaScript が参照している箇所については，書き換えを行わない．その後，pdftk の機能を利用してデータの書き換えにより生じた相互参照表の不整合を修復する．

次にメタデータの墨塗りを図 5 の流れで行う．メタデータの墨塗りには，pdftk の dump_data 機能と update_info 機能を利用する．dump_data は対象 PDF 文書のメタデータをテキスト形式で抽出する機能であり，update_info は対象 PDF のメタデータを更新する機能である．但し，テキスト情報の墨塗りと同様に JavaScript が参照している箇所については墨塗り処理を行わない．

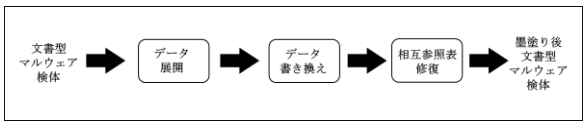


図 4．テキスト墨塗り処理

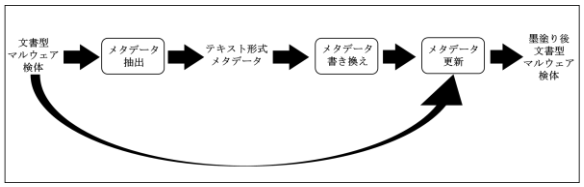


図 5.メタデータ墨塗り処理

# 4　評価実験

本章では実検体を用いた提案手法の評価実験について説明する．D3M Datasets 2010 の文書型マルウェア検体 10 件に対して前章で述べた墨塗り手法を適用し，検体の挙動に影響を与えずに墨塗りが行えるかを確認した．

## 4.1　実験方法

**墨塗り処理**　上述の 10 検体に対して，それぞれテキストとメタデータの全てを削除する墨塗り処理(以降では，全墨塗りと呼ぶ)を行った．さらに 10 検体のうち 6 検体については，API フックログから悪性コードが墨塗り対象コンテンツ情報を参照していることが確認されたため，提案手法を用いて悪性コードが参照していない箇所のみ削除処理を行った(以降では，提案手法墨塗りと呼ぶ)．その後，サンドボックス解析により墨塗り前後の検体の挙動を確認した．

**墨塗り前後の挙動の比較**　文書型マルウェア検体の挙動を変えずに墨塗りを行えたかを検証するために，墨塗り前後の検体文書ファイルのサンドボックス解析を行った．解析システムは図 6 に示す文献[3,4]を参考にした構成となっており，ホスト OS として CentOS 5.7, 犠牲ホストとして VMware Server 1.0.6 を用いて仮想マシンを構築し，OS として Windows XP Professional SP2 を用意した．また, Adobe Reader のバージョン 7.0, 8.0, 9.0，10.0 をインストールしたイメージを用意し，それぞれの場合について検体の挙動を解析した．加えて検体の挙動としてファイル操作，レジストリ操作，プロセス操作を観測するため，Process Monitor[5]をインストールした. 検体ファイルの実行時間(ファイルを Adobe Reader で開いてからの待機時間)は 5 分とした．

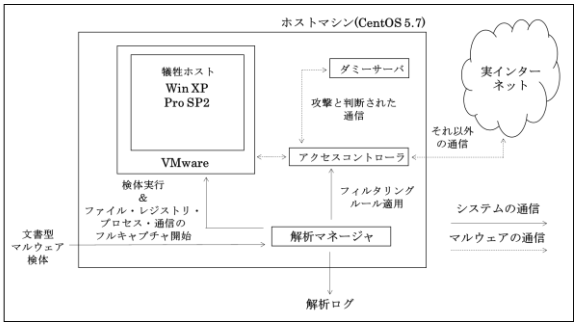


図 6．動的解析環境

## 4.2　実験結果

提案手法による墨塗り処理の前後で文書型マルウェア検体がAdobe Readerで閲覧できる

かを調べた結果を図7に示す．図7の記号「○」は問題なく文書型マルウェア検体を開くことを確認できたものを示しており，記号「×」は開いている途中でAdobe Readerが終了してしまい，文書型マルウェア検体の閲覧ができなかったものを示している．図に示す通り，検体1から検体6まではAcrobat Readerのバージョンに関わらず，墨塗り前後共に問題なく閲覧ができた．検体7と検体9は墨塗り前からAcrobat Reader 7.0と8.0では閲覧ができず，墨塗り後も同様に閲覧ができなかった．

検体8と検体9では，Acrobat Reader 8.0では問題なく閲覧できたものの，それ以外のバージョンでは，Adobe Readerからエラーウィンドウが表示され文書型マルウェア検体の閲覧ができなかった．図7では，このような結果については特に記号「×※」と表記している．

Adobe Readerのバージョン7.0，8.0，9.0，10.0で閲覧時にサンドボックスで観測された検体の挙動をそれぞれ図8～11に示す．なお，これらの図において，ファイル，レジストリ，プロセスの欄における○印は，それぞれファイル操作，レジストリ操作，プロセス操作が観測されたことを示す．特にレジストリの欄において◎印は，○印より多くのレジストリ操作が行われたことを示す．また，通信の欄に記載されている項目はアクセスを試みた通信先のドメインまたはIPアドレスを示す．

まず，図11の通り，Abode Reader 10.0を用いて検体を閲覧した場合，墨塗りの有無にかかわらず，何も不正な挙動が観測されなかった．このことから今回実験に用いた10検体は，Abode Reader 10.0に対しては攻撃を行わない可能性がある．以降では，Abode Reader 10.0以外での閲覧した結果について述べる．

文書型マルウェア検体内のコンテンツを参照する6検体に対して全墨塗りを行った場合，Adobe Reader 7.0では外部との通信が見られなくなり，Adobe Reader 8.0ではレジストリ操作と外部との通信が見られず，Adobe Reader 9.0ではレジストリ操作の一部がなくなり，外部との通信が見られなくなっていた．このことから，全墨塗りにより検体の挙動が変化していると思われる．一方，提案手法の墨塗りの場合は，ファイル操作，レジストリ操作，プロセス操作，通信の観点から見て，顕著な違いは見られず，提案手法の有効性を示しているといえる．

次に検体文書ファイル内のコンテンツの参照が確認できなかった4検体に対して全墨塗りを行った結果，Adobe Reader 7.0と8.0では墨塗り前後で検体の挙動に変化は見られなかった．一方，Adobe Reader 9.0においては，一部の検体についてレジストリ操作が増加したり，通信先が変化したりした．墨塗り処理が影響している可能性があるものの，その原因は不明である．

| 墨塗り前/墨塗り後 | Adobe Reader 7.0 | Adobe Reader 8.0 | Adobe Reader 9.0 | Adobe Reader 10.0 |
|---|---|---|---|---|
| 検体1(MD5:e72f100de9bb68cb2f92b18efe02eb60) | ○/○ | ○/○ | ○/○ | ○/○ |
| 検体2(MD5:bc94c563f2bb16fb5327140015a9b13) | ○/○ | ○/○ | ○/○ | ○/○ |
| 検体3(MD5:23e1b0267e48ca6670f7932c6c75aee4) | ○/○ | ○/○ | ○/○ | ○/○ |
| 検体4(MD5:7cd69d8206bddfd514a5f706fb3eeea7) | ○/○ | ○/○ | ○/○ | ○/○ |
| 検体5(MD5:4010bf8d5cd60e02c69616e0275e65f2) | ○/○ | ○/○ | ○/○ | ○/○ |
| 検体6(MD5:70112cccecdbec307b3b4ae9dd725b8f) | ○/○ | ○/○ | ○/○ | ○/○ |
| 検体7(MD5:37bb84406a56647335db4c441b4780df) | ×/× | ×/× | ○/○ | ○/○ |
| 検体8(MD5:c543031788a0de2514b6a4ed42191125) | ○/×※ | ○/○ | ○/×※ | ○/×※ |
| 検体9(MD5:be30f13a0601ea9b57ae55cc74c8a5f0) | ○/×※ | ○/○ | ○/×※ | ○/×※ |
| 検体10(MD5:374cf5625e2d4ea9f31fd1d304487c63) | ×/× | ×/× | ○/○ | ○/○ |

図 7.提案手法墨塗り前後の文書型マルウェア検体の閲覧可否
※MD5 ハッシュ値は墨塗り前の値

| | Adobe Reader 7.0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 墨塗りなし | | | | 全墨塗り | | | | 提案手法墨塗り | | | |
| | ファイル | レジストリ | プロセス | デフォルト以外の通信 | ファイル | レジストリ | プロセス | 通信先 | ファイル | レジストリ | プロセス | デフォルト以外の通信 |
| 検体1(MD5:e72f100de9bb68cb2f92b18efe02eb60) | × | ○ | × | my contentgude.ru | × | ○ | × | × | × | ○ | × | my contentgude.ru |
| 検体2(MD5:bc94c563f2bb16fb5327140015a9b13) | × | ○ | × | kokojamba.com | × | ○ | × | × | × | ○ | × | kokojamba.com |
| 検体3(MD5:23e1b0267e48ca6670f7932c6c75aee4) | × | ○ | × | toothe.com | × | ○ | × | × | × | ○ | × | toothe.com |
| 検体4(MD5:7cd69d8206bddfd514a5f706fb3eeea7) | × | ○ | × | gogle-analiz.com | × | ○ | × | × | × | ○ | × | gogle-analiz.com |
| 検体5(MD5:4010bf8d5cd60e02c69616e0275e65f2) | × | ○ | × | gogle-analiz.com | × | ○ | × | × | × | ○ | × | gogle-analiz.com |
| 検体6(MD5:70112ccccedbec307b3b4ae9dd725b8f) | × | ○ | × | gogle-analiz.com | × | ○ | × | × | × | ○ | × | gogle-analiz.com |
| | | | | | | | | | | | | |
| 検体7(MD5:37bb84406a56647335db4c441b4780df) | × | ○ | × | myzloy.ru | × | ○ | × | myzloy.ru | | | | |
| 検体8(MD5:c543031788a0de2514b6a4ed42191125) | × | ○ | × | clickgooglo.com | × | ○ | × | clickgooglo.com | | | | |
| 検体9(MD5:be30f13a0601ea9b57ae55cc74c8a5f0) | × | ○ | × | × | × | ○ | × | × | | | | |
| 検体10(MD5:374cf5625e2d4ea9f31fd1d304487c63) | ○ | ○ | ○ | amzinas.com | ○ | ○ | ○ | amzinas.com | | | | |

図 8. Adobe Reader 7.0 の結果

| | Adobe Reader 8.0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 墨塗りなし | | | | 全墨塗り | | | | 提案手法墨塗り | | | |
| | ファイル | レジストリ | プロセス | デフォルト以外の通信 | ファイル | レジストリ | プロセス | 通信先 | ファイル | レジストリ | プロセス | デフォルト以外の通信 |
| 検体1(MD5:e72f100de9bb68cb2f92b18efe02eb60) | × | ○ | × | my contentgude.ru | × | × | × | × | × | ○ | × | my contentgude.ru |
| 検体2(MD5:bc94c563f2bb16fb5327140015a9b13) | × | ○ | × | kokojamba.com | × | × | × | × | × | ○ | × | kokojamba.com |
| 検体3(MD5:23e1b0267e48ca6670f7932c6c75aee4) | × | ○ | × | thoothe.com | × | × | × | × | × | ○ | × | toothe.com |
| 検体4(MD5:7cd69d8206bddfd514a5f706fb3eeea7) | × | ○ | × | gogle-analiz.com | × | × | × | × | × | ○ | × | gogle-analiz.com |
| 検体5(MD5:4010bf8d5cd60e02c69616e0275e65f2) | × | ○ | × | gogle-analiz.com | × | × | × | × | × | ○ | × | gogle-analiz.com |
| 検体6(MD5:70112ccccedbec307b3b4ae9dd725b8f) | × | ○ | × | gogle-analiz.com | × | × | × | × | × | ○ | × | gogle-analiz.com |
| | | | | | | | | | | | | |
| 検体7(MD5:37bb84406a56647335db4c441b4780df) | × | ○ | × | myzloy.ru | × | ○ | × | myzloy.ru | | | | |
| 検体8(MD5:c543031788a0de2514b6a4ed42191125) | × | ○ | × | clickgooglo.com | × | ○ | × | clickgooglo.com | | | | |
| 検体9(MD5:be30f13a0601ea9b57ae55cc74c8a5f0) | × | ○ | × | × | × | ○ | × | × | | | | |
| 検体10(MD5:374cf5625e2d4ea9f31fd1d304487c63) | ○ | ○ | ○ | amzinas.com | ○ | ○ | ○ | amzinas.com | | | | |

図 9. Adobe Reader 8.0 の結果

| | Adobe Reader 9.0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 墨塗りなし | | | | 全墨塗り | | | | 提案手法墨塗り | | | |
| | ファイル | レジストリ | プロセス | デフォルト以外の通信 | ファイル | レジストリ | プロセス | 通信先 | ファイル | レジストリ | プロセス | デフォルト以外の通信 |
| 検体1(MD5:e72f100de9bb68cb2f92b18efe02eb60) | × | ◎ | × | my contentgude.ru | × | ○ | × | × | × | ◎ | × | my contentgude.ru |
| 検体2(MD5:bc94c563f2bb16fb5327140015a9b13) | × | ◎ | × | kokojamba.com | × | ○ | × | × | × | ◎ | × | kokojamba.com |
| 検体3(MD5:23e1b0267e48ca6670f7932c6c75aee4) | × | ◎ | × | toothe.com | × | ○ | × | × | × | ◎ | × | toothe.com |
| 検体4(MD5:7cd69d8206bddfd514a5f706fb3eeea7) | × | ◎ | × | gogle-analiz.com | × | ○ | × | × | × | ◎ | × | gogle-analiz.com |
| 検体5(MD5:4010bf8d5cd60e02c69616e0275e65f2) | × | ◎ | × | gogle-analiz.com | × | ○ | × | × | × | ◎ | × | gogle-analiz.com |
| 検体6(MD5:70112ccccedbec307b3b4ae9dd725b8f) | × | ◎ | × | gogle-analiz.com | × | ○ | × | × | × | ◎ | × | gogle-analiz.com |
| | | | | | | | | | | | | |
| 検体7(MD5:37bb84406a56647335db4c441b4780df) | × | ○ | × | × | × | ○ | × | × | | | | |
| 検体8(MD5:c543031788a0de2514b6a4ed42191125) | × | ○ | × | clickgooglo.com | × | ○ | × | clickgoogle.com | | | | |
| 検体9(MD5:be30f13a0601ea9b57ae55cc74c8a5f0) | × | ○ | × | 213.163.89.54 | × | ○ | × | 213.163.89.54 | | | | |
| 検体10(MD5:374cf5625e2d4ea9f31fd1d304487c63) | × | × | × | × | × | × | × | × | | | | |

図 10. Adobe Reader 9.0 の結果

| | Adobe Reader 10.0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 墨塗りなし | | | | 全墨塗り | | | | 提案手法墨塗り | | | |
| | ファイル | レジストリ | プロセス | デフォルト以外の通信 | ファイル | レジストリ | プロセス | 通信先 | ファイル | レジストリ | プロセス | デフォルト以外の通信 |
| 検体1(MD5:e72f100de9bb68cb2f92b18efe02eb60) | × | × | × | × | × | × | × | × | × | × | × | × |
| 検体2(MD5:bc94c563f2bb16fb5327140015a9b13) | × | × | × | × | × | × | × | × | × | × | × | × |
| 検体3(MD5:23e1b0267e48ca6670f7932c6c75aee4) | × | × | × | × | × | × | × | × | × | × | × | × |
| 検体4(MD5:7cd69d8206bddfd514a5f706fb3eeea7) | × | × | × | × | × | × | × | × | × | × | × | × |
| 検体5(MD5:4010bf8d5cd60e02c69616e0275e65f2) | × | × | × | × | × | × | × | × | × | × | × | × |
| 検体6(MD5:70112ccccedbec307b3b4ae9dd725b8f) | × | × | × | × | × | × | × | × | × | × | × | × |
| | | | | | | | | | | | | |
| 検体7(MD5:37bb84406a56647335db4c441b4780df) | × | × | × | × | × | × | × | × | | | | |
| 検体8(MD5:c543031788a0de2514b6a4ed42191125) | × | × | × | × | × | × | × | × | | | | |
| 検体9(MD5:be30f13a0601ea9b57ae55cc74c8a5f0) | × | × | × | × | × | × | × | × | | | | |
| 検体10(MD5:374cf5625e2d4ea9f31fd1d304487c63) | × | × | × | × | × | × | × | × | | | | |

図 11. Adobe Reader 10.0 の結果

## 4.3 考察

評価実験により，検体の挙動に影響を与えずに墨塗りを行える例が確認できた．今回の実験では，悪性コードが参照する箇所を除いて全てのコンテンツを墨塗りしたが，実際の運用では，どの箇所に墨塗りを行うかを指定できることが望ましい．墨塗りの必要がないコンテンツはそのまま残すことで，より多くの情報を共有できるからである．逆に，墨塗りの必要がある箇所を悪性コードが参照している場合は，今回の手法では墨塗りに失敗する．対策としては，墨塗り処理が必ずしも検体の挙動に影響を及ぼすわけではないため，とりあえず必要な墨塗り処理を行い，その後動的解析により，挙動に変化がないかを確認する試行錯誤による方法が考えられる．また，悪性コードが墨塗り対象コンテンツを参照して行う処理を解析し，同様の実行結果となる別処理に変換する方法が考えられる．

最後に，PDF 形式の文書型マルウェア検体の中には EXE ファイルが組み込まれているものあり，標的型攻撃の場合にはその EXE ファイルの中に標的組織の情報(標的となった組織のプロキシサーバ IP 等)が埋め込まれている．このような情報も秘密情報であり，墨塗りしたい内容である．今回の提案手法で

はそのような埋め込まれた EXE ファイルに対する墨塗りは対象となっていない．今後，PDF 形式の検体だけでなく，Word や Excel といった形式の文書型マルウェアに関しても墨塗りを施す方法について検討したい．

## 5　まとめと今後の課題

PDF 形式の文書型マルウェア検体について挙動を変化させずにテキストやメタデータなどのコンテンツを削除する墨塗り手法を提案した．また，実検体を用いた実験により，提案手法の有効性を確認した．今後の課題は，墨塗り処理の全自動化や，悪性コードから参照されるコンテンツの墨塗りである．

謝辞　本研究の一部は，JSPS 科研費 24680006 の助成により行われた．

## 参考文献

[1] 神薗雅紀,西田雅太,星澤裕二, “動的解析を利用した難読化 JavaScript コード解析システムの実装と評価,” マルウェア対策研究人材育成ワークショップ 2010.
[2] PDFtk - The PDF Toolkit, http://www.pdflabs.com/tools/pdftk-the-pdf-toolkit/ (Last Visit: 2013/008/25)
[3]K. Yoshioka, D. Inoue, Y. Hoshizawa, H. Nogawa, and K. Nakao, ”Malware sandbox analysis for secure observation of vulnerability exploitation”, IEICE Trans. Vol. E92D, No. 5, pp. 955-966, 2009.
[4]K. Yoshioka, and T. Matsumoto “Multi-pass Malware Sandbox Analysis with Controlled Internet Connection”, IEICE Trans. E93A, No.1, pp. 210-218, 2010.
[5] Process Monitor http://technet.microsoft.com/ja-jp/sysinternals/bb896645.aspx（Last Visit:2013/08/25）