時系列データに基づくマルウェア検知アルゴリズムの評価

愛甲 健二† 松木 隆宏†

†株式会社 FFRI
150-0013 東京都渋谷区恵比寿 1 丁目 18 番 18 号 東急不動産恵比寿ビル 4 階
aiko@ffri.jp matsuki@ffri.jp

あらまし 数理統計的手法により実現されたマルウェア検知アルゴリズムは，学習と評価が保持するデータセットに依存するためその有用性の証明が難しい．本稿ではマルウェアの時系列データに着目し，対象となる検知アルゴリズムの性能が持続する期間を調査，推定し，検知精度の低下と再学習が必要となる時期を予測する手法を提案する．

Evaluation of malware detection algorithm by machine learning based on time-series data

Kenji Aiko† Takahiro Matsuki†

FFRI, Inc.
†TokyuFudosanEbisu Building 4F, 1-18-18, Ebisu, Shibuya-ku, Tokyo 150-0013, JAPAN
aiko@ffri.jp matsuki@ffri.jp

Abstract The malware detection algorithm implemented by mathematical statistical methods, proof of its usefulness is difficult because it depends on the data set. In this paper, focusing on time-series data of malwares, we propose a method to investigate the period in which the performance of the detection algorithm of interest is sustained, it is estimated, to predict the timing of re-learning and lowering of the detection accuracy is required.

1 はじめに

マルウェアの増加に伴い，アンチウイルスはシグネチャをベースとした検知手法ではその対応が難しくなり，新たなアプローチの研究，開発が必要になった．特に近年では機械学習，データマイニングをベースとしたマルウェア検知アルゴリズムが多く提案されている．これらの手法は増加する亜種に対応できる一方で，その結果が扱われたデータセットに依存するため，その有用性の証明が難しい．本稿では，この問題を解決する手法のひとつとして時系列データに基づいたマルウェア検知アルゴリズムの評価方法を提案する．

マルウェアは時期ごとに流行や性質が変わるものであり，現時点以降に出現するマルウェアを検知できることが求められる．そのため，収集時期によりマルウェアの類似性にどの程度の違いが出るのかを調べ，その変化から検知アルゴリズムの有用性がどの期間まで保たれるのかを推測する．

マルウェアの類似性に言及した研究は様々

であり，例えば岩村らや大久保らによる機械語命令列、バイナリ列を用いたもの[1][2]，中村らや藤野らによる API コールを用いたもの[3][4]などがある．マルウェアとなるファイルそのものから得られる特徴と，実行後，その実行ログから得られる特徴によって違いはあるだろうが，マルウェアに類似性があることは多くの研究により明らかであろう．

またデータマイニングや機械学習のアルゴリズムを使うことで，マルウェアの解析，クラスタリング，検知アルゴリズムの改善といったものの部分的な自動化を試みた研究も多い[5][6][7]．もちろんマルウェアに限らず，メール，Web 広告のフィルタリングにおいても，これらの技術を用いることによる改善方法が研究されており[8][9][10]，現在，コンピュータセキュリティ全般においてその有効性が試されている．

時系列データに着目した研究としては，柏井ら[11]によるマルウェアの発生過程を統計的に分析する上で時系列分析が用いられた．

本稿では，時系列データを分析し，その結果からマルウェア検知アルゴリズムの有用性を示す手法を提案し，またマルウェアの時系列的な変化について述べる．

2　データセット

本章では，分析に利用するデータの解説と，分析を行う手順について示す．

2.1　概要

対象とするデータセットは約 400,000 検体/月からランダムサンプリングにより選んだ 1000 検体/月とし，期間は 2012/04 から 2013/12 までの 21 ヶ月分を用いた．

サンプリング数は次の式から得た．

$$n \geq \frac{N}{\left(\frac{e}{Z}\right)^2 \frac{N-1}{P(1-P)}+1}$$

母集団のサイズ N，最大誤差 e=0.05，正規分布点 Z=1.96（95%），2.576（99%），母集団の比率 P=0.5 から必要な標本数 n を計算する．95%信頼区間で n>383，99%信頼区間で n>662 により，サンプリング数を 1000 検体/月とした．

全検体は file コマンド（ver5.11）の結果と時間情報をメタデータとして持ち，対象とするデータセットは，file コマンドの結果から次の 2 グループを作成し，いずれも 1000 検体×21 ヶ月分を用意した．

- Win32実行ファイル
- .Netアプリケーション実行ファイル

Win32 実行ファイルは file コマンドによって次の出力が含まれたものとする．

```
PE32 executable (GUI) Intel 80386, for MS Windows
```

また.Net アプリケーション実行ファイルは file コマンドによって次の出力が含まれたものとする．

```
PE32 executable (GUI) Intel 80386 Mono/.Net assembly,
for MS Windows
```

約 400,000 検体/月のうち，Win32 実行ファイルは約 96.3%，.Net アプリケーション実行ファイルは約 2.70%である．

.Net アプリケーション実行ファイルは，.Net や VB といったある特定の種類のマルウェアに限定した場合においても同様の結果になるかどうかを調べるために扱った．

次にデータセットの分布について記述する．

2.2　分布

2012/06，2012/12，2013/06，2013/12 の 4 ヶ月分のマルウェア（400,000 検体/月）の中からランダムサンプリングした各 1000 検体を対象に，k-means を用いて 100 グループにクラスタリングし，マルウェア数の多いクラスタ順に並べたものを図 1 に示す．なお今回クラスタリングに k-means を用いたが，どのようなアルゴリズムを用いても結果に大きな差異はない．

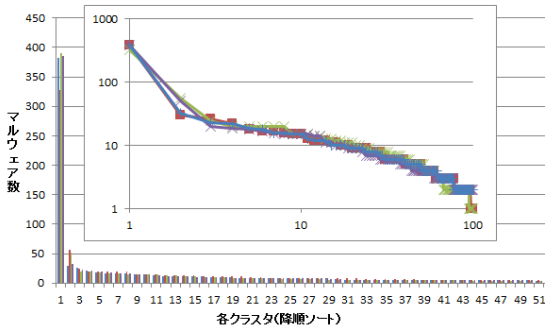


図 1:分布の偏り

図 1 の棒グラフから，どの月においても一部のマルウェア数が極端に多く，それ以降は一気にそのサイズが減る．両対数グラフにすると線形に近い形で減っていることも確認できる．

同時期にランダムに収集されたマルウェアは大半が似たものとなるため，データマイニング，機械学習をはじめとした類似性をベースとして考えられた検知アルゴリズムが有効に機能する．

また，図 1 はスケールフリー性を持つようにも見える．一部のノードが膨大なリンクを持ち，それ以外のノードはいずれもごくわずかなノードとしか繋がっていないネットワーク構造をスケールフリーネットワークと呼ぶ．WWW やインターネットは代表的なスケールフリーネットワークである[12]．図 1 のような，特定の期間に収集されたマルウェアをクラスタリングした場合，各クラスタに属するマルウェア数はこのようにスケールフリー性を持つ場合が多いため，本稿で扱うデータセットもこれに従った．

2.3 手順

類似度の算出に sdhash[13]を用いる．sdhash は ssdeep 同様 Fuzzy hashing のアルゴリズムである．

月ごとに各マルウェア同士の類似度を求め，指定した閾値 s を超える類似度を示すマルウェアがひとつでもあれば真，なければ偽として，真の数をカウントする．x 月をベースとして，x+1, x+2, …, x+20 月において類似したマルウェア数をグラフ化する．

疑似コードを以下に示す．

```
cnt = 0
for(i=0; i < 1000; i++):
  for(j=0; j < 1000; j++):
    d = sdhash(Malware[x][i], Malware[x+N][j])
    if(d > s):
      cnt += 1
      break
print cnt / 1000
```

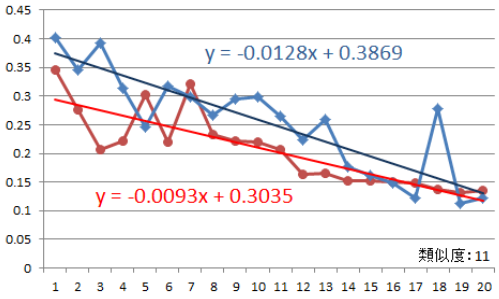
s=11，x=2012/04，N=1 とすると，2012/04 と 2012/05 を比較し，閾値が 11 を超えたものがあれば真としてカウント，真のカウント総数が値となる．マルウェアは各月 1000 検体であるため，最大 1000，最少 0 となる．

3 時系列データ分析

本章では，時系列データの類似性から得られた結果を示す．

3.1 分析結果 1

x=2012/04とそれからNヶ月後(N=1,…,20)とのマルウェアを比較し，グラフ化したものが図2となる．



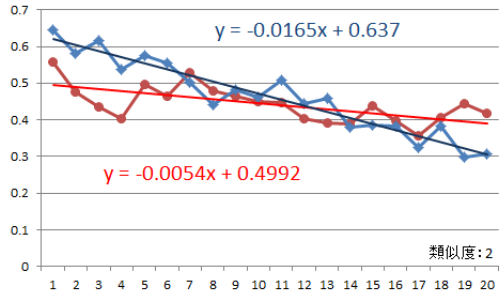


図 2:2012/04 と各月との比較

1 ヶ月後との比較で類似度が高い方(青)が.Net アプリケーション実行ファイル，低い方(赤)が Win32 実行ファイルとなる．また上のグラフが類似の閾値を高くした場合(11)，下のグラフが低くした場合である(2)．閾値の高い方が類似と判断されにくいため，類似するファイル数は減る．

図2から，類似性は時間の経過に従い，線形に落ちていくことが分かる．また閾値を2にした下のグラフでも，1ヶ月後の2012/05との比較では，似たファイルは0.50から0.65程度であり，最初の1ヶ月後の落下がもっとも大きく，その後線形で徐々に落ちていくと考えられる．

3.2　分析結果 2

最初の 1 ヶ月後の落下と閾値の関係を調べるため，次に各月において，閾値を変化させながら 1 ヶ月後との比較を行った結果を図 3 に示す．

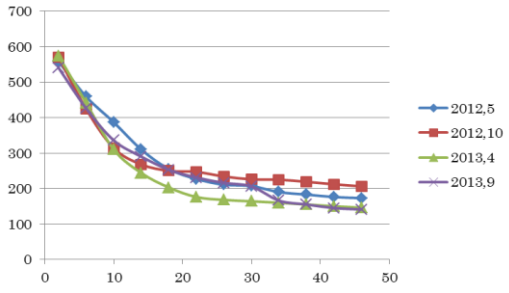


図 3:翌月と閾値(横軸=閾値，縦軸=数)

2012/05，2012/10，2013/04，2013/09を対象に，それぞれ1ヶ月前と比べ，どれだけ類似するものが存在したかをグラフ化した．横軸が閾値，縦軸が類似したものの数である．いずれの月も閾値が低いと500から600ほどになり，閾値を上げていくことで250辺りまで下がる．しかし，それ以降は閾値を上げても緩やかに落ちていくことになる．これは100～200ほどの検体は極めて高い類似性を持っていると推測できる．

3.3　分析結果 3

次に1ヶ月ではなく，複数の月をベースにしたときの，類似度の変化を調べた結果を示す．

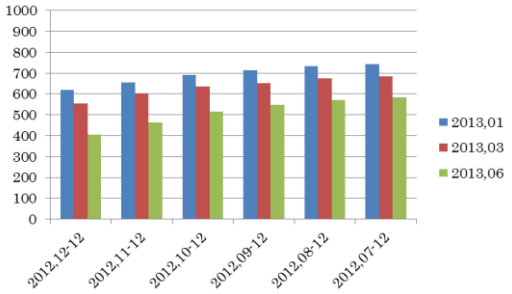


図 4:複数の月をベースにした場合の変化

左端のグループは2012/12のみと，2013/01，2013/03，2013/06をそれぞれ比べたときの結果である．2013/01で600超程度，2013/06で400超程度類似するものがあった．左から2番目のグループは，2012/11と2012/12を合わせたデータ2000検体と2013/01，2013/03，2013/06をそれぞれ比べたときの結果である．いずれの月も2012/12単体に比べ，30～50程度の増加が見られる。このようにして6ヶ月分まで遡ってグラフにしたものが図4である．

そして，月を増やすごとにどれだけ検知数が増加したのかを図 5 に示す．

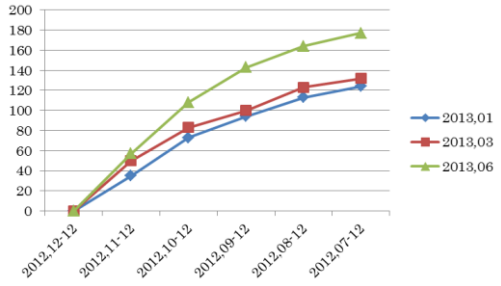


図 5:検知数の増加

2012/12を0とし，そこからどれだけ増加したのかを図 5 に示した．月を増やすごとに類似と判断された数は増加するが，徐々に増加数が減っていることが分かる．

以上から，期間を増やすことで改善が見込めるが，比例して類似するマルウェアが増えるわけではないことが分かる．

4　API ログの時系列データ分析

3章ではファイルに対してsdhashを用いて類似度を算出し，その結果から時系列分析を行った．本章では，API Callログに対して同様のことを行い，その結果を示す．

4.1　データセット

データセットは，約 400,000 検体/月からランダムサンプリングにより選んだ約400～1100検体/月とし，期間は2012/04から2013/12までの21ヶ月分を用いた．これらは基本的に3章で用いたものと同じであるが，環境によって実行できない検体などがあり，ランダムサンプリングされたすべての検体から必ずしも有用な API Call ログを取得できないため，対象とする検体数は月ごとにばらけることになった．

実行ログは Cuckoo Sandbox(ver0.6)で取得し，その API Call ログのみを扱った．

4.2　手順

基本的に 3 章と同様だが，類似度の算出にレーベンシュタイン距離を用いる．API Call ログを繋げた文字列に対して，レーベンシュタイン距離を算出して類似度を求めた．

```
cnt = 0
for(i=0; i < len(x); i++):
  for(j=0; j < len(x+N); j++):
    d = Levenshtein(MalAPIs[x][i], MalAPIs[x+N][j])
    if(d > s):
      cnt += 1
      break
print cnt / len(x+N)
```

4.3　分析結果

取得したデータを次に示す．

100 API

Cnt	Total	Rate
357	465	0.768
699	941	0.743
719	1031	0.697
640	968	0.661
660	1011	0.653
710	1012	0.702
671	948	0.708
459	679	0.676
457	756	0.604
390	616	0.633
510	874	0.584
555	891	0.623
584	868	0.673
493	855	0.577
423	818	0.517
367	713	0.515
247	516	0.479
282	545	0.517
218	481	0.453
385	787	0.489

300 API

Cnt	Total	Rate
318	465	0.684
581	941	0.617
630	1031	0.611
513	968	0.53
525	1011	0.519
511	1012	0.505
508	948	0.536
313	679	0.461
378	756	0.5
302	616	0.49
420	874	0.481
430	891	0.483
476	868	0.548
346	855	0.405
321	818	0.392
253	713	0.355
195	516	0.378
212	545	0.389
182	481	0.378
298	787	0.379

表 1:API Call の時系列

左の表が先頭 100API を用いたもの，右の表が先頭300APIを用いたものである．

それぞれの表は右から類似した数，総数，割合の順序で並んでおり，1 行目から 1 ヶ月後との比較，2 ヶ月後との比較，3 ヶ月後との比較と続く．よって 2012/04 をベースにして，2012/05 との比較が1行目，2012/06との比較が2行目となる．

表1をグラフにすると，次になる．

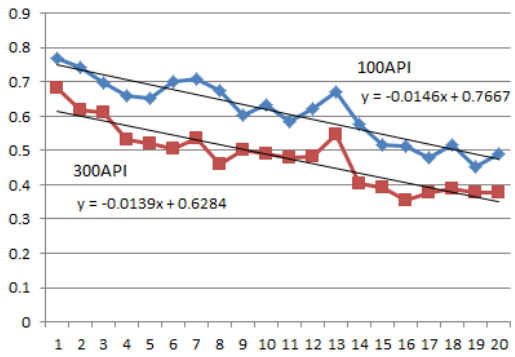


図 6:API Call ログの時系列

ファイルに対し，sdhash を用いて類似度を算出した場合と似たグラフになることが確認できる．以上から，マルウェアの実行により得られた特徴に対しても，同様の理論が成り立つと推測できる．

5　結論

調査結果により，x 月と x+1 月のマルウェアでは，類似するものの割合が大きく減る．また x+1 月以降は線形で安定して割合が減っていくことが分かる．便宜上，ここからは前者を初期低下，後者を継続低下と呼ぶ．

数理統計的手法を用いたマルウェア検知アルゴリズムで重要なことは，未来に出現する脅威を識別できることにある．同じ時間に存在したマルウェア群を 2 つに分け，片方を学習，片方を評価に使用して得た結果は，本質的に未来のマルウェアを検知できる証明にならない．

その有用性を示したいとき，時系列情報をメタデータとして持っているマルウェア検体を x 月，x+1 月という隣り合う 2 ヶ月で集め，x 月を学習データ，x+1 月を評価データとして扱い，その結果を確認することで，アルゴリズムの評価と検知率の予測ができる．

大量の時系列データがある場合はよいが，仮になくとも，隣り合う 2 ヶ月を使うことで初期低下を調べることはでき，そこから線形に落ちていくことを考慮することで将来的な検知率をある程度予想できる．6 ヶ月から 12 ヶ月分の時系列データがあれば継続低下における「傾き」を調べることができ，より正確に検知アルゴリズムの評価が可能になる．

また，図 4 より，6 ヶ月にまたがるマルウェア検体を用いることで 100～200 程度の改善がみられた．直近の月だけでなく，6～12 ヶ月分のマルウェアを用いることで性能の上昇が見込めるが，図 5 より，増加値は比例ではなく，長い期間になればなるほどに増加値は減った．これは期間を長くとることで性能が上がるが，長くとりすぎるほど徐々に改善が見込めなくなることを意味する．

最後に API Call ログに対して同様の調査を行い，その結果も同じになることが確認できた．これにより，ファイルから得られる特徴に対しても，実行ログから得られる特徴に対しても同様に，時系列データに基づいた性能の推測が可能であることを確認した．

6　まとめ

本稿では，マルウェアの時系列データから，時間の経過によってネットワークに存在するマルウェアがどの程度変化するのかを示し，それを利用して，マルウェア検知アルゴリズムの有用性の評価を行う提案手法を示した．

またマルウェアには，ファイルから得られる特徴でも，API Call ログから得られる特徴でも同様に類似性があることを述べ，それらを特徴として用いた検知アルゴリズムが有用に機能することを述べた．

今後の研究として，時系列データから各クラスタに所属するマルウェアの増減を可視化，予測し，近い未来に流行するクラスタ，新しく作られるクラスタの予測ができないかといった時系列分析による予測について調査していく．

7　参考文献

[1] M. Iwamura, M. Itoh, and Y. Muraoka, “Automatic Malware Classification System Based on Similarity of Machine Code Instructions”, CSS, 2010

[2] R. Okubo, M. Morii, R. Isawa, D. Inoue, and K. Nakao , “Function Estimation Method for Malwares based on part of Binary Code”, CSS, 2012
[3] R. Nakamura, R. Matsumiya, K. Takahashi, and Y. Oyama, “Identification of Subspecific Malware by Utilizing Kullback-Leibler Divergences”, CSS, 2013
[4] A. Fujino, and T. Mori, “Analysis of Massive Amount of API Call Logs Collected from Automated Dynamic Malware Analysis Systems”, CSS, 2013
[5] U.Bayer, P.M. Comparetti, C.H.C. Kruegel, and E. Kirda. “Scalable, behavior-based malware clustering”, NDSS, 2009
[6] K. Rieck, P. Trinius, C.Willems, and T.Holz, “Automatic analysis of malware behavior using machine learning”, Journal of Computer Security, 19(4), 2011
[7] A. Kantchelian, S. Afroz, L. Huang, A. C. Islam, B. Miller, M. C. Tschantz, R. Greenstadt, A. D. Joseph, and J.D. Tygar , “Approaches to Adversarial Drift”, ACM, 2013
[8] H.Lee, and A.Ng, “Spam deobfuscation using a hidden markov model”, In proceedings of the Second Conference on Email and Anti-Spam, 2005
[9] A. Kantchelian, J.Ma, L.Huang, S.Afroz, A.D. Joseph, J.D. Tygar. “Robust detection of comment span using entropy rate”, AISEC 2012. ACM, 2012
[10] Z. Li, K. Zhang, Y. Xie, F.Yu, and X. Wang, “Knowing your enemy: Understanding and detecting malicious web advertising”, CCS, 2012
[11] Y. Kashii, M. Morii, D. Inoue, and K. Nakao, “Time Series Analysis of Malware Using NONSTOP Data”, CSS, 2013
[12] Barabási, Albert-László and . Albert, “Emergence of scaling in random networks”, Science, 1999
[13] Vassil Roussev, “Scalable Data Correlation”, IFIP, 2012