複数の解析環境から取得したマルウェアの振る舞い情報の非類似性尺度に関する検討

林 直樹† 重本 倫宏† 鬼頭 哲郎† 仲小路 博史†

†株式会社日立製作所
244-0817 横浜市戸塚区吉田町２９２番地
{naoki.hayashi.vr, tomohiro.shigemoto.jh,
tetsuro.kito.bs hirofumi.nakakoji.vt,}@hitachi.com

あらまし 近年，攻撃者が意図した特定の環境以外では動作しないマルウェアが出現しており，解析が困難化している．そのようなマルウェアを機械的，且つ高精度に解析する方式として，我々は多種の解析環境を用いて動的解析を行い，解析結果を対比して分析する方式を検討している．単一の解析結果に対する距離尺度としては，例えば編集距離などを用いる研究があるが，複数種別の動的解析結果を複合的に用いた分析方式を検討した研究は少ない．そこで，本検討では，単一種別の解析結果に導入される距離を量的変数として用いて構成した多次元尺度を提案する．さらに，MWSデータセット2014を用いて計算量を評価し，実際上計算可能であることを示す．

A Study on Dissimilarity Measure for Malware's Behaviors from Multiple Sandboxes

Naoki Hayashi† Tomohiro Shigemoto†
Tetsuro Kito† Hirofumi Nakakoji†

†Hitachi, Ltd.
292, Yoshida-cho, Totsuka-ku, Yokohama-shi, Kanagawa-ku, 244-0817 Japan
{naoki.hayashi.vr, tomohiro.shigemoto.jh,
tetsuro.kito.bs, hirofumi.nakakoji.vt}@hitachi.com

Abstract Recently malware with anti analysis function has emerged. So, we are developing a system that performs dynamic analysis using multiple sandboxes at parallel, each of which has different environment. Distance measures for the result of a single analysis such as edit distance have been already studied, but few studies are done on a combination of multiple dynamic analysis results. In this paper, we consider a multidimensional scale based on the metrics on single type analysis result. And we verify the computational feasibility of that by use of the MWS2014 data sets.

1　はじめに

近年，組織を狙った標的型攻撃は組織化，ツール化されてきている．攻撃者はマルウェアの生成ツールを用いることで，攻撃に用いるための新種のマルウェアを容易に生成することが出来るため，攻撃対象にされる組織には，そのような新種のマルウェアが日々多数届く状況にある．

これらの新種のマルウェアは，マルウェアのハッシュ値やバイナリの部分列など，スタティックな情報がベースになっているシグニチャを用いるアンチウィルスソフトでは検知することが困難であり，攻撃を素通ししてしまう．しかしながら，新種の被疑検体が組織に届くたびに，全てを人手で解析した上で，それがマルウェアであった場合にはシグニチャを生成する，といった対策を採ることも，全ての被疑検体に完全に対応することは工数的に殆ど実現不可能である．このように，攻撃者側の攻撃手法は既にスケールしており，組織型のマルウェア解析と検知の技術にもスケール性が求められている状況にある．

また，近年のマルウェアには，動的解析によるマルウェアの発見，解析を困難化することを目的に，動作環境が特定の条件に一致すれば動作を停止させる“解析回避機能”を具えたものが出現してきている．

これにより，単純に一つの動作環境を準備しただけでは，マルウェアが“正常に動作する”（＝攻撃活動を行う）とは限らず，従って分析に必要なマルウェアの振る舞い情報を取得できない可能性がある．

上記の状況を解決すべく，我々は複数の動的解析エンジンを連携させて，高精度に被疑検体の振る舞いを分析する「多種環境動的解析方式（M3AS: Multi Modal Malware Analysis System）」を検討している．本稿では，多種環境動的解析方式の概要と，多種環境動的解析方式から取得される複数の振る舞い情報を用いた多次元尺度の構成方法について我々の検討結果を記載する．

本稿の構成は次の通りである．まず 2 章に，多種環境動的解析方式の概要と，本稿で取り上げる課題を述べる．次に，3 章に単一の動作環境から得られた振る舞い情報に対して定義される距離尺度について既知の方法をまとめる．次に 4 章に多種環境動的解析方式の処理の中で得られる複数の振る舞い情報を用いた多次元尺度の構成法について述べる．5 章で MWS2014 データセットを用いた評価結果を述べる．6 章と 7 章は結果の考察とまとめである．

2　多種環境動的解析方式(M3AS)

マルウェア解析をスケールさせる方法としては，解析処理を機械化して自動化することが一つの有効な方法である．中でも動的解析は，バイナリの難読化の有無などに依らず，マルウェアを動作させさえすれば振る舞いを抽出できるため，機械化に適している．

しかしながら，近年のマルウェアには，解析の困難化を目的にして，動作環境が特定の条件に一致した場合には自らの動作を停止させる“解析回避機能”を具えたものが出現してきている．そのような条件としては，具体的には例えば，動的解析エンジンがサンドボックス内で動作させる特定プロセスの存在有無の確認や，自らが対象とする脆弱性を持ったアプリケーションがインストールされていない場合，など様々である．

そこでまず我々は，下記の二点を主眼にして，被疑検体を動作させるためのサンドボックスの構成要素として望ましいものを洗い出し，約 100 種類のサンドボックスを構成した[1]：

1. 企業や政府組織で利用されているOSやアプリケーションであること．
2. 脆弱性が公開されていたり，既知のマルウェアによって狙われたことがあるアプリケーションであること．

我々が検討している多種環境動的解析方式は，このようにして構築した多数のそれぞれOS

やアプリケーション構成が異なる解析環境全てを用いて被疑検体を動的解析にかけ，それぞれの解析環境から得られた振る舞い情報間の相関や，過去に解析した他の被疑検体の振る舞い情報との相関について分析することで，被疑検体の特徴的な振る舞いを明らかにし，また，解析回避動作に起因した構成要素を特定することを目的としている(**エラー! 参照元が見つかりません。**).

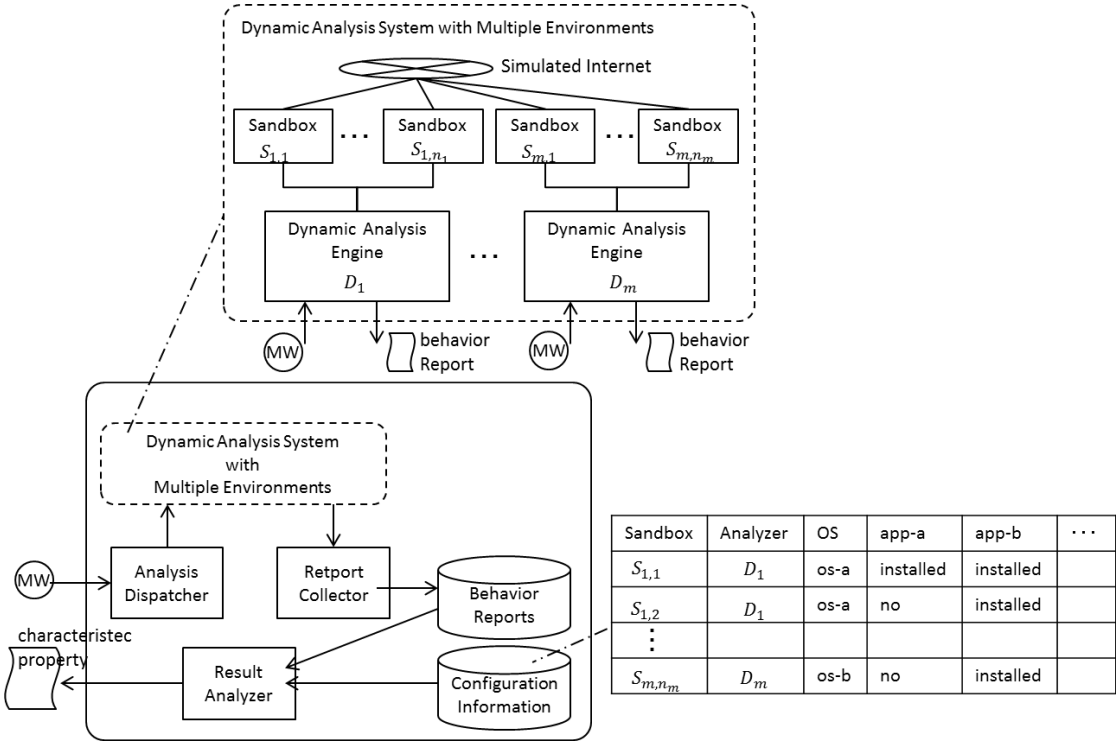

図 1. 多種環境動的解析方式(M3AS)の概要

多種環境動的解析方式では，企業や政府組織でよく利用されている環境や，脆弱性などが判明している環境を複数用いて解析を行うため，動作条件が合致しないような環境では動作しないマルウェアの振る舞い情報も取得することができる[1].

本稿の以下の章においては，多種環境動的解析方式の中でも特に，過去に解析した他の被疑検体の振る舞い情報との相関について分析するための方式について述べる.

なお，過去の振る舞い情報を用いて分類する目的は，被疑検体が，既にマルウェアだと判っているマルウェアとの振る舞いと類似している場合に，被疑検体の悪性判定として活用できる点などがある. また，被疑検体を人手で詳細に解析する際，解析を効率化するための補助情報（例えば既知の類似検体の難読化手法など）としても活用することが出来る.

しかしながら，本方式から取得される振る舞い情報は非常にデータ量が非常に大きい. 単一種の動的解析結果が数 MB～数十 MB であるため，100 種の振る舞い情報を併せると場合によっては1GB近くになる. 数千検体分のそのようなサイズのデータを実運用可能な速度で分類することは必ずしも容易ではない. また，そもそも複数種の振る舞い情報を複合的に分析する方式の検討自体，既存研究は少ない.

そこで，本稿の以降の章では，単一種の振る舞い情報に対する分類方式に関する既存研究を踏まえた上で，複数種の振る舞い情報を分析する方式について検討し，さらに計算量の評価を行う.

3　振る舞い情報の距離尺度

単一の動的解析結果を基にマルウェアを分類する手法については様々に研究がなされている[2-4,6]. これらは基本的に, 振る舞い情報に対して何らかの距離尺度を定義し, その定義に従って各マルウェアの振る舞い同士の距離を計算することでクラスタリングを行っている.

振る舞い情報に対する距離尺度の定義は様々に成し得るが, 例えば, [3]によると, マルウェア振る舞いに対する距離尺度としては, 少なくとも下記の三点を要求すべきである:

1. 似通った振る舞いでは距離が小さく, 逆に異なる振る舞いでは距離が大きいこと.
2. 計算機的に実際的に計算可能であること.
3. 近いスケールでのシステムコールの並びが異なる場合は距離に大きく反映されることが望ましく, 逆に, 広い区間でのシステムコールの順序の相違は距離にあまり影響を与えるべきではない. これは, 単一スレッドについての振る舞いの相違については距離に反映させ, 別スレッドに関する相違については距離にあまり反映させるべきではない, という理由によるものである.

[3]では, 上記のような条件を考慮した上で, 編集距離, 近似編集距離, 正規化圧縮距離, Trie Tree 構造上の距離の 4 種の距離についてそれぞれ考察を述べている. 本検討における, 複数環境から得られる振る舞い情報の距離尺度についても, 上記[3]で考察されている 4 種の距離を基本にして検討を行った.

まず, 上記 4 種の距離尺度についてそれぞれ概要を述べる.

3.1　編集距離(LD)

編集距離(Levenshtein Distance)は, 連続データの非類似度を測るための距離尺度としてよく用いられる距離である. なお, マルウェアの振る舞い情報の分析にこの距離尺度を導入した先行研究としては, 例えば[2]がある.

通常の編集距離は, 二つの文字列がどの程度異なっているかを比較する尺度の一つであり, 片方の文字列を一文字ずつ挿入・削除・置換していき, 最小で何回の操作でもう片方の文字列に変換できるかを距離として用いるものである.

この計算は動的計画法で効率的に解けるクラスの問題であることが知られており[7], 計算量は$O(n \cdot m)$ (ただし, n,m はそれぞれの文字列の長さ)である.

本検討で対象とするデータは単なる文字列ではなく, System Call 情報を要素とする順序データである. 従って, 編集距離を計算する際には, System Call の名称を一つの塊(アルファベット)として取り扱うこととする. 即ち, 図 2 に示すように System Call を特定の文字に one-to-one で置換するような同型変換を前処理として行う.

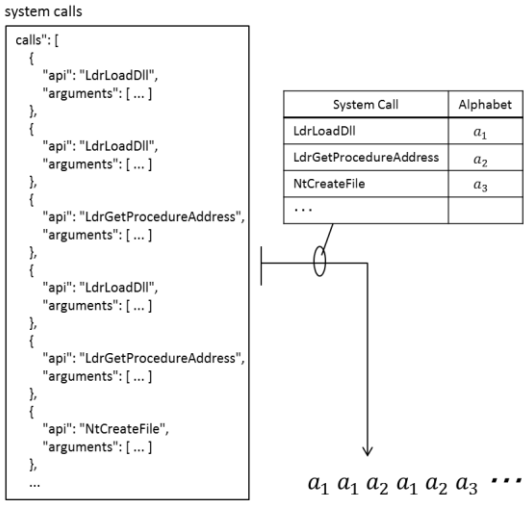

図 2. System Call のアルファベット化

3.2　Fuzzy Hashing

3.1 に述べた編集距離は, System Call が一つでも異なれば距離が 1 加算される. 同じマルウェアでも動作させるタイミングによっては

System Call の順序や数が多少変化する動的解析の結果に対して適用するには，少し過敏すぎる懸念がある．また，計算オーダが文字列長の積になっていることも，場合によっては数万の数の System Call が現れる振る舞い情報を扱う上では問題となり得る．

それらを緩和するために好適な距離関数の一つとしては，Fuzzy Hashing[8]が挙げられる．

Fuzzy Hashing は Context Triggered Piecewise Hashing (CTPH) とも呼ばれるものであり，rolling hash と piecewise hashing を組み合わせたものである．まず対象データを rolling hash で計算し，rolling hash の結果が事前に定めた条件に合致した箇所を境界として piecewise hashing でハッシュ値を計算して出力する．

Fuzzy Hash は大部分が一致し，一部だけが異なるファイルを比較するような用途に適している．また，出力されるハッシュ値は元のデータと比較して十分に短いため，非類似度を計算するための編集距離の計算に掛かる計算コストも大幅に減少させることが期待できる．

Fuzzy Hash を用いた非類似度の計算は，出力されたハッシュ値の編集距離を計算することで行う．従って，System Call 情報を要素とするデータの比較を行うためには，まず，3.1 節の場合と同様の図 2のアルファベット化を行った上で，Fuzzy Hash 値を計算し，編集距離を計算することになる．

なお，5 章に述べる評価においては，Fuzzy Hash の実装として ssdeep[5]を用いた．

3.3 正規化圧縮距離(NCD)

正規化圧縮距離 (Normalized Compression Distance) は，Kolmogorov 複雑度の考え方に基づく距離尺度であり，圧縮効率の改善度を用いて2つのデータ列の比類似性の尺度とするものである[9]．具体的には，圧縮関数を C，比較するデータ列をそれぞれ x, y，$|\cdot|$をビット長としたとき，次の計算式で定義される．

$$NCD(x,y)=\frac{|C(xy)|-\min\{|C(x)|,|C(y)|\}}{\max\{|C(x)|,|C(y)|\}},$$

（ただし，$NCD(x,x)=0$）

NCD は圧縮関数を用いるため，どのような形式のデータに対しても用いることができることが利点である．

なお，5 章に述べる評価においては，圧縮関数 C の実装として gzip を用いた．

3.4 n-gram Trie Tree 上の距離(TTD)

Trie Tree とは，単語集合を保持することに向いたデータ構造である．本稿ではデータ列の n-gram を単語セットとして用いて，対象データ列の Trie Tree を構成する．

本稿の n-gram Trie Tree はノードに単語パターンの出現回数と単語の終端か否か，エッジにアルファベットを保持する(図 3)．

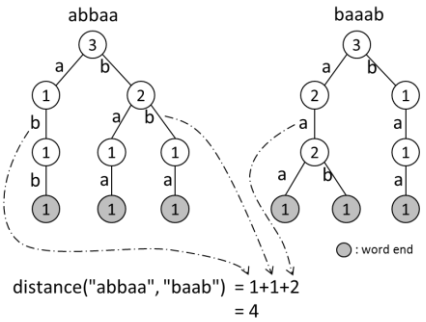

図 3. 3-gram Trie Tree 構造と距離定義

Trie Tree構造上の距離尺度は次のような計算処理によって定義付けられるものであり，非類似性の尺度である[10,11]．

- Step 0： 距離を 0 とする
- Step 1： 互いの木についてルートノードから同一のアルファベットが割り当てられたエッジをトラバースする．
- Step 2： 片方にしか存在しないエッジが現れた場合，そのエッジの先の部分木のルートノードが保持する出現回数を距離に加算する．

- Step 3：Step 1, Step 2 を全ての共通する枝を調べ終わるまで続け，終わった時点の距離を最終的な距離とする．

この尺度の計算量は$O(k|x|+k|y|)$（ただし，k は x,y に共通する単語の最大長）であることが知られている[10]．本稿においては，単語として n-gram を用いるため単語長は固定であり，従って計算量はデータ長に対して線形オーダである．

4　多次元尺度の構成

多種環境動的解析方式から得られる複数の振る舞い情報から，マルウェア間の非類似性を計算する方式について検討する．

すなわち，複数種別の振る舞い情報を用いて定義されるマルウェア間の類似性を$\hat{d}$とすると，$\hat{d}(\mathrm{MW}^s,\mathrm{MW}^t)=\varphi(\mathbf{b}^s,\mathbf{b}^t;d)$，

ただし$\mathbf{b}^u=\begin{pmatrix}b^u_{1,1} & \cdots & b^u_{1,1}\end{pmatrix}^{\mathrm{T}}$，d は単一種別の振る舞いに対する距離関数，なる$\varphi$について検討する．

φには少なくとも次を要求すべきである[図 4]．

1. 0 以上の実数値であり，同一のデータに対しては距離が 0 であること．すなわち，

   $\varphi(\mathbf{b}^s,\mathbf{b}^t;d)\geq 0,(0 if\ s=t)$

   動的解析の結果は実行のたびに多少ノイズが載るため，$\hat{d}(\mathrm{MW}^s,\mathrm{MW}^s)=0$ は要求しないが，$\hat{d}(\mathrm{MW}^s,\mathrm{MW}^s)\approx 0$ である必要はある．
2. 可換性が成立すること．すなわち，

   $\varphi(\mathbf{b}^s,\mathbf{b}^t;d)=\varphi(\mathbf{b}^t,\mathbf{b}^s;d)$ が成立．

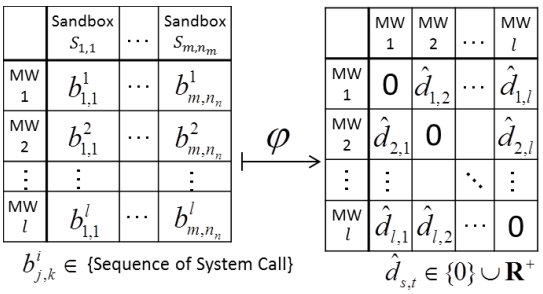

図 4. 非類似性の多次元尺度

また，本検討で用いる尺度としては下記の要求を満たすことが妥当である．

3. ある動作環境$S_{j,k}$において，2つの検体 $\mathrm{MW}^s,\mathrm{MW}^t$ の振る舞い情報の非類似性が十分小さく（即ち，$d(b^s_{j,k},b^t_{j,k})\approx 0$），かつ，それらの振る舞い情報がデータとして一定程度の大きさがある場合には，$\hat{d}(\mathrm{MW}^s,\mathrm{MW}^t)$も十分小さい値にする．

   これは，対解析回避機能を具える検体などの場合に，少なくとも一つの環境で類似動作が見られた場合（ただし，全く動作しなかった場合を除く）の非類似性を小さくする，という理由である．

上記の条件を満たすφとしては，下記が挙げられる．

$$\varphi(\mathbf{b}^s,\mathbf{b}^t;d)=\min_{j,k}\left(d(b^s_{j,k},b^t_{j,k})\right)$$

5　評価

MWS2014 データセットの Cuckoo 及び Yarai Analyzer Professional（各 3000 検体）の2種類の動的解析結果を用い，4 章で述べた

多次元尺度で定義した距離行列を計算することで評価を行った．単一種の振る舞い情報に対する距離関数 d としては，3 章に述べた，編集距離(LD)，Fuzzy Hashing(ssdeep)，正規化圧縮距離(NCD)，n-gram Trie Tree 構造上の距離(TTD)のそれぞれを用いた．また，NCD の圧縮関数としてはgzipを用い，TTDとしては n-gram としては，[3]の中で最も計算時間とクラスタリング結果のバランスが良いと結論付けられている3-gramを用いることとした．

なお，実験に用いた計算機のスペックは OS: MicroSoft® Windows7 64bit，CPU：Intel® Core i7-3770，メモリ32GBであり，プログラムは Python 2.7.6 を用いて実装した．

計算処理に掛かった時間を表 1に示す．

表 1. 多次元尺度の計算時間

距離関数 d	時間(秒)	備考
LD	5,068,050	200 検体分の処理時間(22,425秒)からの推定値
ssdeep	279	-
NCD	276,398	200 検体分の処理時間(1,223秒)からの推定値
TTD	12,124	-

6 考察

検証ではデータセットとして提供されている振る舞い情報の種別が 2 種であったが，我々が検討している多種環境動的解析方式では 100 種程度の振る舞い情報を用いることを想定している．

提案の多次元尺度は種別数に対して線形オーダであるため，振る舞いの種別を100種類に増加させた場合，単純にはおよそ 50 倍程度に時間が増加することを見込む必要がある．

以下，仮に 1 日平均 20 の新しいマルウェアが組織に届くとした場合，実運用可能か否かを評価する．マルウェアに感染したことに組織が気付くまでに最長で 2 年を越えることもあるという報告がある[12]ため，3 年分の検体の振る舞い情報を保持する必要があると考えると，約 22,000 検体の振る舞い情報を用いた距離行列を構成する必要がある．

新規検体が届いた場合に距離行列を更新するためには，すでに届いている検体と新しい検体との距離を計算すればよいため，22000 の要素を新規に計算することになる．

表 1.の計算は実際には ${}_{3000}C_2 = 44985$ （C はコンビネーション）要素分の計算を行っていることを考えると，距離行列の更新に掛かる時間は単純には表 2.の通りである．

表 2. 距離行列の更新に掛かる計算時間

距離関数 d	時間(秒)
LD	24,785
ssdeep	1
NCD	1,351
TTD	59

1日に 20 回の更新処理を行うことが出来るのは，ssdeep と TTD を用いた場合のみであり，LD や NCD を用いた尺度は実用的であると言えない．この要因としては，振る舞い情報に含まれる System Call の数が Cuckoo では平均 34,978，最大 315,918，yarai professional では平均 68,197，最大 517,485 と非常に多いためであると思われる．

すなわち，多種環境動的解析方式で用いる多次元尺度としては，振る舞い情報のデータ長に対して線形オーダであるか，もしくは振る舞い情報を事前に十分に短くする必要があり，本検討の中では，ssdeep，もしくは TTD を用いた距離尺度が実用的に利用可能であると言える．

7 まとめ

解析回避機能を具えるマルウェアを解析するアプローチとして，多種解析動的環境方式を提案した．また，当該方式によって取得される複数種の振る舞い情報に対し，既に単一種の振る舞い情報に対する既知の距離関数を用いて

多次元尺度を構成した．さらに，構成した多次元尺度を実装し，MWS データセット 2014 を用いて計算量的に計算可能であるか検証し，ssdeep や 3-gram Trie Tree 構造を距離関数として用いた場合に提案した多次元尺度が計算可能であることを確認した．

今後の課題としては，本手法による振る舞いのクラスタリング結果と静的解析による分類結果の比較，入力データとして System Call 以外の情報も含めた場合の評価，が挙げられる．

参考文献

[1] 仲小路 博史, 他, “進化する標的型攻撃に対抗するマルウェア自動解析技術”, 日立評論 2014 年 3 月号,
http://www.hitachihyoron.com/2014/03/pdf/03a12.pdf

[2] T.Lee and J.J.Mody, “Behavioral classification”, in Proc. of EICAR2006, 2006,

[3] Martin Apel, et.al, “Measuring Similarity of Malware Behavior”, SICK2009, 2009,

[4] 藤野 朗稚, 森 達哉, “自動化されたマルウェア動的解析システムで収集した大量 API コールログの分析”, MWS2013, 2013,

[5] “Fuzzy hashing and ssdeep”, [Online]
http://ssdeep.sourceforge.net/

[6] M.Bailey, et.al, “Automated classification and analysis of internet malware”, proc. of RAID 2007, 2007,

[7] D.S.Hirschberg, “A linear space algorithm for computing maximal common subsequences”, Communications of the ACM, vol.18, no. 6, p.341-343, 1975,

[8] Jesse Kornblum, “Fuzzy Hashing” , http://www.dfrws.org/2006/proceedings/12-Kornblum-pres.pdf

[9] Rudi Cilibrasi, Paul M. B. Vitanyi, “Clustering by compression”, IEEE TIT (2005),

[10] Konrad Rieck, Pavel Laskov, “Linear-Time Computation of Similarity Measures for Sequential Data”, Journal of Machine Learning Research 9(2008), 2008,

[11] Konrad Rieck, Pavel Laskov, “Detecting Unknown Network Attacks Using Language Models”, DIMVA2006, 2006,

[12] TREND MICRO, “セキュリティマガジン TREND PARK” [Online], http://www.trendmicro.co.jp/jp/trendpark/coretech-threatintelligence/martin/20130819062637.html