# プロファイル型ホワイトリスト準最適化手法の提案

重本倫宏[1,2] 川口信隆[1] 藤井翔太[1] 藤井康広[1] 西田昌平[1] 菊池浩明[2]

**概要：** 近年，高度化されたマルウェアによるサイバー攻撃の被害が増加している．これらの脅威による被害の拡大を防ぐためには，組織に侵入した攻撃者を迅速に検知し，対処することが重要となる．組織に侵入した攻撃者を検知する手法として，通常動作や定常業務をプロファイル化し，これに該当しない活動を攻撃として検知するプロファイル型ホワイトリストを用いた対策が存在する．しかし，ホワイトリストを適切に管理しないと，誤検知や，検知見逃しが発生する場合がある．本報告では，誤検知リスクと検知見逃しリスクの観点からホワイトリストを最適化する手法を提案する．さらに，プロトタイプを用いた評価実験により，既存手法と比較し，ホワイトリストの質を 15%以上向上できることを示す．

# Proposal of Profile Based White List Semi Optimization method

TOMOHIRO SHIGEMOTO[1,2] NOBUTAKA KAWAGUCHI[1] SHOTA FUJII[1]
YASUHIRO FUJII[1] SHOHEI NISHIDA[1] HIROAKI KIKUCHI[2]

## 1. はじめに

近年，民間企業や，防衛関連企業，公的機関を狙ったサイバー攻撃が顕在化しており，個人，企業，国家の利益や安全性を損なうリスクが高まっている．特に APT（Advance Persistent Threat）攻撃[1]は，秘密裏に，そして執拗に長期間攻撃を続ける点で従来の脅威とは異なり，マルウェアの侵入を検知あるいは防止することは不可能になりつつある．たとえば，2015 年 6 月に日本年金機構において，遠隔操作型マルウェアに感染した職員の端末から基礎年金番号を含む個人情報が約 125 万件漏えいし，大きな社会問題となった[2]．これらの脅威に対抗するためには，組織に侵入した攻撃者を迅速に検知し，対処することが重要となる．

組織に侵入した攻撃者を検知する手法として，ブラックリスト型およびホワイトリスト型の対策がある．本稿では，ホワイトリスト型対策の中でも特に，適用先環境の通常動作や定常業務を学習してリスト化することで，これに該当しない活動を攻撃として検知する，プロファイル型ホワイトリストを用いた対策に着目する．プロファイル型ホワイトリストは，ブラックリストや画一的なホワイトリストと比べ，未知のマルウェアや正規ツールを悪用した攻撃を検知できるという利点がある．

プロファイル型ホワイトリストにより攻撃者を検知するには，適切に管理されたホワイトリストを用いなければならない．たとえば，ホワイトリストに登録する活動を制限しすぎると誤検知が発生してしまう．一方で，ホワイトリストに登録する活動を増やしすぎると検知見逃しが発生してしまう．また，業務や端末構成が変化した際には，ホワイトリストを学習し直さなければならないが，いつ再学習するのが望ましいのか定量的な指標が存在しない．

そこで本稿では，誤検知リスクと検知見逃しリスクの観点からホワイトリストを最適化する手法を提案する．また，運用期間中にホワイトリストの再学習が必要になるタイミングを判断する手順について検討する．さらに，ある組織の同一部署に属する 26 台の端末の 2 ヶ月間の活動ログを用いた評価実験によって，本方式は既存手法と比較し，ホワイトリストの質を 15%以上向上できることを確認する．

まず，2 章では関連研究と本方式の位置付けについて説明し，3 章でプロファイル型ホワイトリストの課題について述べ，4 章でプロファイル型ホワイトリスト準最適化手法を提案する．5 章で実環境を用いた評価実験および最適なパラメータの検討を行い，6 章を本稿のまとめとする．

## 2. 関連研究

既存研究として，プロセスや通信に基づくホワイトリストを用いて標的型攻撃やマルウェアを検知する技術が提案されている．中里ら[3]は，日常で利用しているプロセスか否かをプロセス情報から判断し，不審なプロセスを特定する手法を提案している．Takemori ら[4]は，ユーザが端末を操作していない時間帯に発生した通信をホワイトリストと比較することで，ボットを検知する手法を提案している．報告者らのグループでも，複数端末で行われる様々な種類の不審活動を分析し，攻撃者の拡散経路をグラフ構造として抽出することで拡散活動を検知する手法を提案している[5]．しかしこれらの方式は，誤検知リスクや検知見逃しリスクを考慮したホワイトリスト作成方法については議論されていない．

時系列データを扱う機械学習の分野では, Sliding Window

1 （株）日立製作所
Hitachi Ltd.
2 明治大学
Meiji University

を用いて定期的にモデルを更新する方式が検討されている[6]．提案手法でも定期的に再学習を実施する点で Sliding Window に近い方法を採用している．しかし提案手法は，再学習の前にその必要性を確認することで，不必要な再学習に伴う演算負荷を低減する点に特徴がある．

また近年，学習時及び運用時における，機械学習モデル自体に対する攻撃技術が，Adversarial Machine Learning として注目されている[7]．学習時における主な攻撃手段は，学習データに攻撃情報を混入させるものである．提案手法では，学習期間の直前に初観測された活動に対して大きな検知見逃しリスクを付与することで，この問題を解決している．

提案手法の特徴は，誤検知リスクと検知見逃しリスクの観点からホワイトリストを最適化する点にある．提案手法を用いることで，既存方式で活用されているホワイトリストの質を向上させることが可能となる．

## 3. プロファイル型ホワイトリストの課題

### 3.1 プロファイル型ホワイトリストによる対策

プロファイル型ホワイトリストを用いた対策では，システム導入後の一定期間(たとえば一ヶ月)を学習期間とし，この学習期間中に観測された活動（プロセス起動や，内部通信等）は通常業務での活動とみなされ，ホワイトリストに登録される．運用期間中は，ホワイトリストに登録されていない活動を不審活動とみなし，不審活動を検出した際には，管理者へ通知したり，当該活動をブロックしたりする．プロファイル型ホワイトリストを用いたサイバー攻撃検知の例を図 1，図 2 を用いて説明する．図 1 は，遠隔操作型マルウェアを用いた攻撃の流れを例示したものである．まず攻撃者は，標的としている組織に，遠隔操作型マルウェアを添付したメールを送付する．組織のユーザが誤って添付ファイルを実行すると，遠隔操作型マルウェアに感染してしまう．組織に侵入した遠隔操作型マルウェアは，感染端末を操作し，組織内で感染を拡大し，最終的に情報を窃取する．

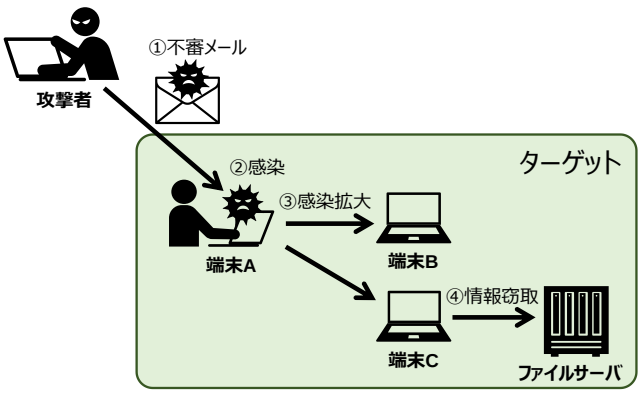


図 1　サイバー攻撃の流れ
Figure 1　Overview of Cyber Attack.

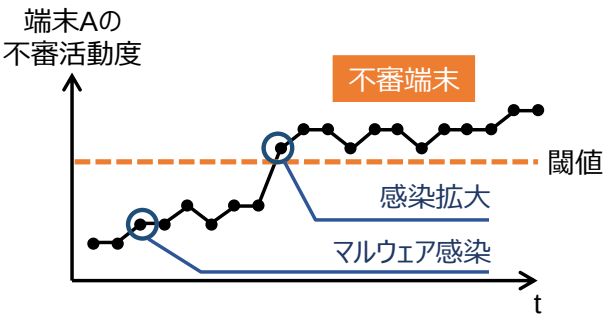


図 2　プロファイル型ホワイトリストによる検知
Figure 2　Detection of Profile Based White List.

この時，端末 A では，攻撃者の行う活動（マルウェアプロセスの起動や感染拡大によって生じる普段通信を行わない端末への通信）はホワイトリストに登録されていない活動であるため，端末 A の不審活動度（不審活動の頻度）が上昇する（図 2）．不審活動度がある一定の閾値を超えた場合に，当該端末が攻撃者に遠隔操作されているとみなし，管理者へアラートを通知する．

### 3.2 プロファイル型ホワイトリストの課題

プロファイル型ホワイトリストを用いた対策の精度を向上させるためには，ホワイトリストの質が重要となる．図 3 を用いてホワイトリスト作成の課題を説明する．例えば，既に攻撃者が端末内に侵入していることに気づかずに，学習期間に行われた活動をもとにホワイトリストを作成することを考える．

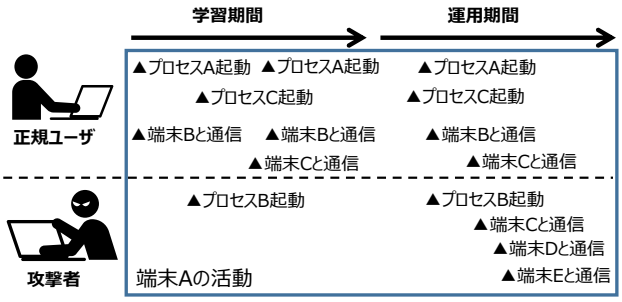


図 3　端末 A の活動
Figure 3　Activity of PC A.

ホワイトリストの作成方法として，学習期間に行われた活動すべてを登録した場合，ホワイトリストには，「プロセス A 起動」，「プロセス B 起動」，「プロセス C 起動」，「端末 B と通信」，「端末 C と通信」が登録される．しかし，このホワイトリストには，学習期間に発生していた攻撃者の活動「プロセス B 起動」が含まれており，当該ホワイトリストを用いて運用を行った場合，運用期間中に発生した攻撃者の活動（「プロセス B 起動」）を不審活動として特定することができない．さらに，運用期間中に発生した攻撃者の

活動「端末 C と通信」も，学習期間中の正規ユーザの活動としてホワイトリストに含まれているため，不審活動として特定することができない．

一方，ホワイトリストに登録する活動を厳選し，例えば「プロセス A 起動」,「端末 B と通信」の 2 つを登録した場合，運用期間中に発生した正規ユーザの活動「プロセス C 起動」を不審活動として特定してしまう．

本稿で定義する理想的なホワイトリストとは，正規ユーザの活動を網羅しつつも，学習期間中の攻撃者の活動や，将来攻撃に利用される活動を排除したものである．図 3 の例では，「プロセス A 起動」,「プロセス C 起動」,「端末 B と通信」から構成されるホワイトリストが，理想的なホワイトリストとなる．しかし，学習期間中の攻撃者の活動や，将来攻撃に利用される活動を予め特定して排除することは不可能である．

また，ホワイトリスト運用の課題として，ホワイトリストを更新する時期が不明確であることが挙げられる．例えば，正規ユーザが新しいソフトウェアを導入し，利用を開始した場合，ホワイトリストの更新が行われていなければ，当該ソフトウェアを起動するたびに不審活動として特定してしまう．

以上述べたように，プロファイル型ホワイトリストの作成・運用の課題は，大別すると 3 点にまとめられる．

**【課題 1】誤検知が発生する**

ホワイトリストに登録する活動を必要以上に制限しすぎた場合，運用期間中に学習に含まれない正規の活動が行われた場合に，不審な活動と検知してしまう．

**【課題 2】検知見逃しが発生する**

学習期間中に発生した活動を無制限に登録しすぎた場合，運用期間中に発生した攻撃が偶然ホワイトリストに含まれてしまい，検知できない可能性がある（課題 2-1）．

また，学習期間中に攻撃が発生していた場合は，当該攻撃による活動がホワイトリストに混入してしまい，運用期間中に同様の攻撃が発生した際に，検知できない可能性がある（課題 2-2）．

**【課題 3】ホワイトリストの質が劣化する**

運用期間中にホワイトイリストを再学習するべきタイミングがわからず，端末の役割や構成の経年変化に，うまく追随できない．

## 4. ホワイトリスト準最適化手法の提案

### 4.1 提案手法の概要

3.2 節で述べた課題を解決するため，プロファイル型ホワイトリスト準最適化手法を提案する．各課題へのアプローチ方法を以下に示す．

【課題 1】【課題 2】への対応として，誤検知リスク・検知見逃しリスクの観点からホワイトリストの質を定量評価する評価指標を策定する（アプローチ 1）．さらに，与えられた活動ログ及び策定した評価指標を基に，最適あるいは準最適なホワイトリストを，遺伝的アルゴリズムを用いて探索する方式を確立する（アプローチ 2）．

【課題 3】への対応として，策定した評価指標を基に，運用中に再学習が必要なタイミングを判断する手順を確立する（アプローチ 3）．

本手法の全体像を図 4 に示す．

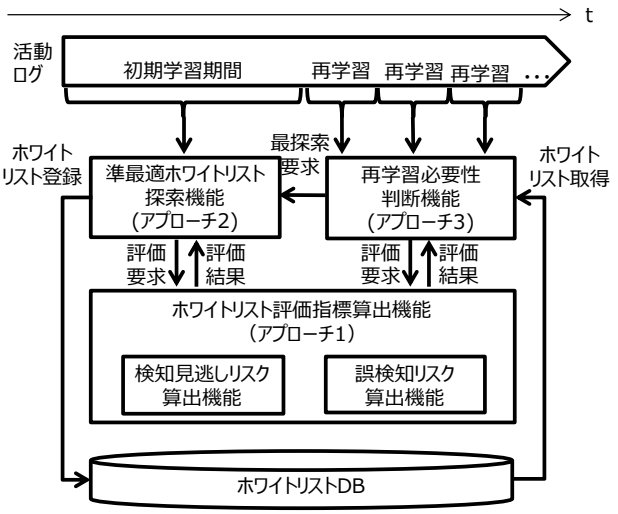


図 4　提案手法の全体像

Figure 4　Overview of Approach.

プロファイル型ホワイトリストにおける，誤検知リスクと検知見逃しリスクはトレードオフの関係にある．これは IDS の Precision（精度）と Recall（再現率）の関係と同様である．このため，IDS の評価では，Precision と Recall の調和平均である F-measure[8]を，両要素を統合した評価指標として用いる．F-measure により，精度・再現率が異なる複数の IDS の中で，どれがより優れたものか判断出来る．（アプローチ 1）においても同様に，統合的な評価指標（WL_index）を定めることで，プロファイル型ホワイトリストの質を定量化する．

また，ホワイトリストには数百のエントリが含まれるため，エントリのどのような組合せが，評価指標を最適化するのかは自明ではない．このため，（アプローチ２）では，遺伝的アルゴリズムを用い，ある活動ログが与えられた場合の WL_index の値を最適化あるいは準最適化するホワイトリストを作成する方式を確立する．

理論上，（アプローチ１）（アプローチ２）を用いて運用中に再学習を行うたびに，ホワイトリストは最適化あるいは準最適化される．しかし，機械学習には演算コストが伴うため，不要な再学習の頻発は避けるべきである．このため，（アプローチ３）として，WL_index に基づき，運用中で，再学習が必要なタイミングを判断する手順を確立する．これにより不要な再学習を削減できる．また，多数の端末

のホワイトリストをメンテナンスする必要がある場合は，どの端末のリストから再学習するべきか，優先度付けができる．

### 4.2 提案手法の詳細

#### 4.2.1 評価指標（WL_index）策定

**(1) 評価指標（WL_index）**

ある端末 h のホワイトリストを作成する時刻を T とし，T までの学習期間中に活動ログ $L_h$ が得られているとする．また，$L_h$ には $C=\{c_1,c_2,\cdots,c_M\}$ の M 種類の活動が記録されている．$L_h$ を基に作成されたプロファイル型ホワイトリスト $W_h$ には，$W_h=\{e_1,e_2,\cdots,e_S\}$ の S 個（$S \leqq M$）のエントリが掲載される．

ここで，$L_h$ からは，学習期間中に発見された攻撃に起因するログは，予め除去されているとする．ただし，潜在的に $L_h$ 内に未発見の攻撃が残存している可能性はある．

誤検知リスクを定量化した値を $f(W_h,L_h,T)$，検知見逃しリスクを定量化した値を $g(W_h,L_h,T)$とする．このとき，$W_h$ の WL_index を，トレードオフの関係にある $1-f(W_h,L_h,T)$と $1-g(W_h,L_h,T)$の調和平均として表現する．WL_index を定式化すると，

$$\mathrm{index}(W_h,L_h,T) = 1-\frac{(1+\theta)\cdot\{1-f(W_h,L_h,T)\}\cdot\{1-g(W_h,L_h,T)\}}{\theta\cdot\{1-g(W_h,L_h,T)\}+\{1-f(W_h,L_h,T)\}} \quad (1)$$

となる．Index が低い程，ホワイトリストの質が高い．$0\leq index(),f(),g()\leq 1$ である．なお，$\theta$ は，検知見逃しリスクに対して誤検知リスクをどの程度重要視するかを示すパラメータであり，$\theta \geq 0$ である．

**(2) 誤検知発生リスク**

$f(W_h,L_h,T)$は，エントリが無い空のホワイトリスト $W_{null}$ と $W_h$ を比較したときの誤検知発生スコアの比として算出する．誤検知発生スコアは，ホワイトリストをある活動ログに適用した場合に発生する誤検知頻度を定量化したものである．即ち，$f(W_h,L_h,T)$は，

$$f(W_h,L_h,T)=\frac{f_s(W_h,L_h,T)}{f_s(W_{null},L_h,T)} \quad (2)$$

として求められる．$f_s(W_h,L_h,T)$は，$W_h$ を $L_h$ に適用した場合の誤検知発生スコアである．

誤検知発生スコアは，拡散活動検知システム[5]のアルゴリズムに習い，ある window 時間（たとえば一時間）の間に発生した，ホワイトリストに含まれない活動のユニーク数に基づくペナルティとして求める．よって，$f_s(W_h,L_h,T)$を定式化すると

$$f_s(W_h,L_h,T)=\int_0^T p(u(W_h,L_h,t-window,t),TH_{max},d)dt \quad (3)$$

となる．$u(W_h,L_h,t\text{-}window,t)$は，期間[t-window, t]で発生する活動のうち，$W_h$ に含まれないもののユニーク数を返す関数である．

$p(u,TH_{max},d)$は，関数 u が返すユニーク数に応じたペナルティを求める関数であり，以下のとおり定義される．

$$p(u,TH_{max},d)=\begin{cases} u^d, & u<TH_{max} \\ TH_{max}^d, & u\geq TH_{max} \end{cases} \quad (4)$$

d は，ユニーク数に対するペナルティの重み付けであり $d\geq 0$ である．$TH_{max}$ は，ペナルティが極端に高い値をとるのを防ぐためのガイドである．

ここで，$L_h$ に攻撃に起因する活動群 $C_a$ が含まれており，かつ $W_h$ には登録されていない場合，$f_s(W_h,L_h,T)$は真の値 $f_s(W_h,L_h/C_a,T)$よりも大きな値となる．$f_s(W_h,L_h/C_a,T)$を実際に求めるのは不可能である．ここで，

A) $L_h$ と比較して $C_a$ が十分小さい

B) $C_a$ のうち $W_h$ に登録されている要素の発生頻度の比率と，$L_h$ のうち $W_h$ に登録されている要素の発生頻度の比率に大差が無い

の何れかが成り立つのであれば，$f(W_h,L_h/C_a,T)\fallingdotseq f(W_h,L_h,T)$ となり，$f(W_h,L_h,T)$は真の値とほぼ等しくなると考えてよい．一般に攻撃の活動量が多い程，事前に発見される機会は多くなる．このため，攻撃が含まれる場合でも(A)が成り立つ蓋然性は高い．また，ホワイトリスト作成方式は $C_a$ 内の要素を区別なく選択するため，(B)が成り立つ蓋然性も高い．本研究では(A)または(B)が成り立つことを前提とする．

**(3) 検知見逃しリスク**

$g(W_h,L_h,T)$は，すべての C の全要素を含むホワイトリスト $W_{full}$ と比較したときの，$W_h$ の見逃しスコア $g_s(W_h,L_h,T)$ の相対値として算出する．即ち，$g(W_h,L_h,T)$は，

$$g(W_h,L_h,T)=\frac{g_s(W_h,L_h,T)}{g_s(W_{full},L_h,T)} \quad (5)$$

となる．

$W_h$ の見逃しスコア $g_s(W_h,L_h,T)$は，$W_h$ を適用した場合に発生しうる誤検知頻度を定量化した値である．$g_s(W_h,L_h,T)$はホワイトリスト内の各エントリ e の見逃しスコア $g_{se}(e,T)$ の総和として算出する．すなわち，

$$g_s(W_h,L_h,T)=\sum_{i=1}^{i=S} g_{se}(e_i,T) \quad (6)$$

となる．

ここで，課題 2-1 より，どのようなエントリであっても，運用中に発生した攻撃の一部が偶然一致するリスク（検知見逃しリスク I）は必ず存在する．よってリスク値は一様に 1 と設定する．

ただし，本来は悪意ある活動でなく正常業務でも発生するが，標的型サイバー攻撃に悪用されやすい活動も存在する．例えば JPCERT/CC では，標的型攻撃に悪用されやすい Windows コマンドを公表している[9]．そこで，外部のナレッジを基に，標的型攻撃に悪用されやすい活動群 $B_{set}$ を収集し，これに含まれる活動に対しては例外的なリスク値を

付与する．

一方で，課題 2-2 に対しては，一般に，作成日時（即ち T）の直前に発生した活動をホワイトリストに加えるのは望ましくない．これは，未だ発見されていない攻撃に起因する活動がリストに混入する恐れがあるためである．例えば，Verizon のレポート[9][10]では，標的型攻撃が発生してから発見されるまでに，ワーストケースでは数ヶ月程度を要するとされている．そこで，ある活動が実際には攻撃に起因するものであるリスク（検知見逃しリスク II）を，活動の初観測日時と T の時間差に半比例すると仮定する．これは，攻撃発生から時間が経過するほど，何らかの手段で攻撃を発見する機会が多くなるためである．リスト作成前に攻撃が発見されれば，攻撃に起因する活動はリストの候補から外すことができる．

WL_index では，課題 2 に起因するリスクを，$B_{set}$ 及び T と初観測日時の差を引数する減衰関数で表現する．1 日前に新規観測された活動をホワイトリストに登録するリスク値を R, 30 日後のリスク値を 1，$B_{set}$ に含まれる活動へのリスク値を $R^B$ とすると，検知見逃しリスク I 及び II を統合したリスク値 $g_{se}(e_i,T)$は

$$g_{se}(e_i,T) = \begin{cases} R^B, & e_i \in B_{set} \\ R^{\frac{30-(T-e_i^{first})}{29}}, & e_i \notin B_{set}, T-e_i^{first} < 30days \\ 1, & e_i \notin B_{set}, T-e_i^{first} \geq 30days \end{cases} \quad (7)$$

として算出できる．なお，$e_i^{first}$ は，エントリ内の活動が最初に発生した時刻を示す．

以上より，任意のホワイトリストについて WL_index を計算できる．

#### 4.2.2 準最適ホワイトリスト作成

$L_h$ からホワイトリストを作成する場合，考えられる組み合わせは $2^M$-1 通りある．通信・プロセス起動を監視対象の活動とした場合，実際には M>100 となるため，全ての組み合わせを試行して，WL_index が最小となるホワイトリストを特定するのは計算量の点で困難である．

そこで遺伝的アルゴリズム[12]を用いて WL_index を最小化するホワイトリストの近似解を探索する方式を検討する．遺伝的アルゴリズムは，ナップサック問題・巡回セールスマン問題など，多項式時間で最適解を算出するのが難しい NP 困難な問題に対して，近似解（準最適解）を導くことができる．

遺伝的アルゴリズムでは，データ（解の候補）を，複数の「遺伝子」で構成される「染色体」として表現する．そして，複数のランダムな染色体をベースに，任意の評価関数において評価値が高い個体を優先的に選択して，交叉（組み換え）・突然変異などの遺伝子操作を繰り返し世代交代しながら，解（＝評価値が高い染色体）を探索する．

提案手法では，ある活動 $c_i$ をホワイトリストに含めるか否かを，遺伝子 $d_i$ で表現する．$d_i$=1 なら $c_i$ はリストに含まれ，$d_i$=0 なら $c_i$ はリストに含まれない．ホワイトリストに含まれる活動の組合せは，染色体 D={$d_1$,$d_2$,…,$d_M$}として扱う．染色体の遺伝子操作手順としては，選択・交差・突然変異を含む一般的なアプローチを用いる．

図 5 に，ホワイトリスト作成方法の擬似コード（generateWhiteList）を示す．まず，関数 initialize は，乱数を基に一定数（例えば 500）の初期染色体のセットを作成する．関数 G は $D_n^g$,$L_h$ で指定されたホワイトリストを作成する．次に，現世代で最も WL_index の評価値が低い染色体を選択する．P 世代前と比べて WL_index に大きな差が無い場合，これ以上の進化は無いと考え，アルゴリズムを打ち切る．アルゴリズムが打ち切られない場合，関数 computeNextGeneration は，現世代を基に，選択・交叉・突然変異により次世代の染色体を作成し，探索を継続する．

```
1   generateWhiteList(L_h, T){
2     //initialize the variable for current generation
3     g=0
4     //the number of seeds
5     N=500
6     //generate N initial seeds for 0th generation
7     D^0={D_1^0, D_2^0, …, D_N^0}=initialize(L_h, N)
8
9     While( true ){
10      //function G returns a white list based on D_n^g, L_h
11       //The following two lines of codes find the best
12       //WL_index among the current population
13      best^g = min_{1≤n≤N} index(G(D_n^g, L_h), L_h, T)
14      S = arg min_{1≤n≤N} index(G(D_n^g, L_h), L_h, T)
15
16      //if the index is not improved compared with
17      //the past P generation, return the best value
18      if(best^{g-P} − best^g ≤ δ)
19        return G(D_s^g, L_h) as W_h
20
21      //compute the next generation from the current
22      //population by using selection, cross over
23      //and mutation
24      D^{g+1}=computeNextGeneration(D^g)
25      g++
26    }
27  }
```

図 5　ホワイトリスト作成方法の擬似コード

Figure 5　Pseudo Code of White List Generation.

#### 4.2.3 再学習必要性判断

本節では，$W_h$ 作成後の再学習手順について述べる．本手順では，ホワイトリストを作成した時刻 T 以後，インターバル I(例えば I=7days)ごとに，経年変化に伴う $W_h$ の質の変化 Q を評価する．Q がある閾値より高い場合，ホワイトリ

スト作成後，端末 h の挙動に大きな変化があったと判断して，$W_h$ を再学習対象とする．

時刻 T から，k 回のインターバル経った時刻 $T^k$ (=T+kI) までの期間に発生したログを $L_h^k$ と表記する．このとき，$Q(W_h,L_h^k,T^k)$は

$$Q(W_h, L_h^k, T^K) = \frac{\mathrm{index}(W_h, L_h^k, T^k)}{\mathrm{index}(W_h, L_h^1, T^1)} \quad (8)$$

と評価される．$T^1$ はホワイトリストを作成してから最初のインターバルである．当然，$Q(W_h,L_h^1,T^1)=1$ となる．Q の値域は，Q>0 である．Q は時間軸で必ずしも単調増加ではない．たとえば，運用中，全く誤検知が発生しない場合，$T^k>T^{k-1}$ より，

$$\mathrm{index}(W_h, L_h^k, T^k) < \mathrm{index}(W_h, L_h^{k-1}, T^{k-1}) \quad (9)$$

となる．

そして，閾値$Q_{TH}$に対して

$$Q(W_h, L_h^k, T^K) \geq Q_{TH} \quad (10)$$

が成り立つとき，ホワイトリスト $W_h$ を更新する．

## 5. 評価実験

### 5.1 実験概要

提案方式の評価実験について述べる．実験では，ある組織内ネットワークに設置された同一事業部に属する PC から活動ログを取得し，評価指標及びホワイトリスト作成方式について評価した．学習用のログは，実験に参加している 26 台の端末の通信・プロセス起動ログであり，学習期間は 2017 年 6 月 16 日から 2017 年 7 月 10 日までである．またホワイトリスト作成日時は 2017 年 7 月 11 日の 00:00:00 とした．

評価では，2017 年 7 月 11 日から 2017 年 8 月 7 日までの運用期間に際して，誤検知が発生する期間（誤検知期間）を測定する．誤検知期間は，運用期間中で，ホワイトリストに含まれないユニークな活動が window 時間当たり閾値 TH 個以上発生する期間の長さである．学習期間・評価期間を合わせ，活動ログの総レコード数は 3,652,514 件である．

表 1 に，評価に用いたパラメータのデフォルト値を，表 2 に遺伝的アルゴリズムのパラメータを示す．

表 1　パラメータのデフォルト値

Table 1　Default Parameters.

| パラメータ | デフォルト値 |
|---|---|
| TH | 4 |
| $\theta$ | 10 |
| d | 1 |
| $B_{set}$ | $\phi$ |
| $TH_{max}$ | 10 |
| R | 10 |
| Window | 1hour |
| I | 7days |

表 2　遺伝的アルゴリズムのパラメータ

Table 2　Genetic Algorithm Parameters.

| パラメータ | 値 |
|---|---|
| 交叉率 | 0.65 |
| 突然変異率 | 0.1 |
| 染色体の数 | 500 |
| 種を保存する割合 | 0.1 |

WL_index およびホワイトリスト作成方式の性能評価のために，それぞれについて，比較対象方式を設定する．

WL_index の比較対象とする評価指標(alt_index)では，個々の活動の発生回数と発生時刻を基にホワイトリストの質を評価する．具体的には，誤検知発生リスク $f(W_h,L_h,T)$の代わりに，$W_h$ 内の活動が $L_h$ で出現する頻度 $r(W_h,L_h)$ を用いる．定式化すると以下のとおりとなる．関数 count は，引数で与えられたエントリの出現回数を求める．

$$\mathrm{alt}_{\mathrm{index}(W_h,L_h,T)} = 1 - \frac{(1+\delta)\cdot \mathrm{r}(W_h, L_h)\cdot\{1-\mathrm{g}(W_h, L_h, T)\}}{\delta\cdot\{1-\mathrm{g}(W_h, L_h, T)\} + \mathrm{r}(W_h, L_h)} \quad (11)$$

$$\mathrm{r}(W_h, L_h) = \frac{\sum_{e\in W_h} count(e, L_h)}{|L_h|} \quad (12)$$

WL_index と alt_index の比較を通し，ホワイトリストに含まれない活動の発生頻度を考慮することで，WL_index が相対的に優れた評価指標となることを示す．

ホワイトリスト作成方式の比較対象としては，遺伝的アルゴリズムに依らず発生頻度を基にホワイトリストを作成する貪欲法（greedy 方式）を用いる．greedy 方式では，検知見逃しリスクが特定の値 M を超えない範囲で，学習期間中の発生頻度が多い活動から順番にホワイトリスト $W'_h$ に登録する．提案方式と greedy 方式の比較を通し，遺伝的アルゴリズムを用いることで相対的に優れたホワイトリストを作成できることを示す．

評価用プログラムは全て Java JDK1.7 で実装した．ステップ数は約 2000 である．また評価に用いた PC のスペックは，OS=Windows 7 64bit Home Premium, CPU=Corei7-2600, Memory=4GB である．

### 5.2 評価結果

#### (1) 学習コスト

図 6 に，遺伝的アルゴリズムを適用したときの，ある端末の WL_index の学習状況を示す．30 世代程度で値がほぼ収束していることがわかる．

表 3 に，$W_h$ の作成及び $Q(W_h,L_h^1,T^1)$の算出にかかる所要時間を示す．ただし，演算開始時点で，活動ログおよびホワイトリストのデータがメインメモリ上にあることを前提とする．$W_h$ の作成には約 85 秒，$Q(W_h,L_h^1,T^1)$の算出には約 0.2 秒かかっており，Q 値算出の方が約 400 倍速いことがわかる．仮に 1 万台の端末のホワイトリストを管理す

る場合，全端末のホワイトリストの作成には約 236 時間かかる．一方，Q 値算出にかかる時間は 30 分程度である．このため，前述のとおり，再学習時には，一度に全ホワイトリストを作成するのではなく先ず Q 値を求めたうえで必要に応じて作成するのが好ましいといえる．

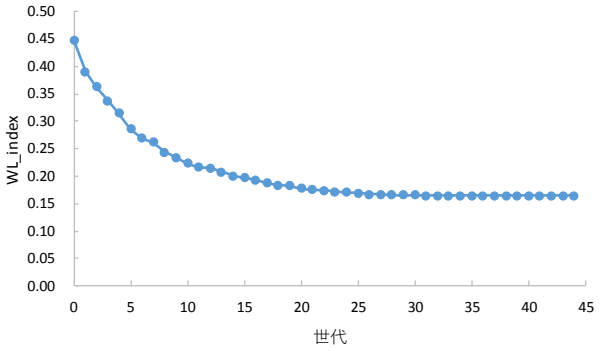


図 6　WL_index の更新過程
Figure 6　Update Process of WL_index.

表 3　ホワイトリスト作成及び Q 値算出の所要時間
Table 3　Time for Generating Whitelist and Computing Q value.

| | ホワイトリスト作成時間[msec] | Q 値算出時間[msec] |
|---|---|---|
| 平均 | 85,402.58 | 198.27 |
| 標準偏差 | 22,935.54 | 42.99 |

### (2) 誤検知と検知見逃しリスクの関係

図 7 に，$\theta$ を 1 から 1000 まで変化させた場合の，$W_h$ の誤検知期間と検知見逃しリスクのトレードオフを示す．各値は 26 端末の平均値である．$\theta$ を大きくする，すなわち誤検知を減らす方向に WL_index をチューニングするほど，誤検知期間は低減し，検知見逃しリスクは増加する．

alt-index を基にした $W'_h$ と比較すると，$W_h$ の性能曲線はより原点側に位置する．即ち，検知見逃しリスクが同一である $W_h$ と $W'_h$ がある場合，$W_h$ の方が誤検知期間は短くなる．以上より，運用中の誤検知期間を短縮するという目的に対しては，WL_index は alt-index に比べて優れた評価指標といえる．

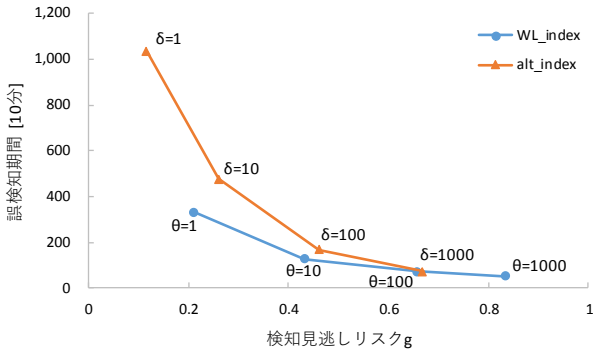


図 7　指標評価の比較
Figure 7　Comparison of Evaluation Criteria.

図 8 に，提案方式で作成した $W_h$ と greedy 方式で作成した $W'_h$ との比較を示す．両方式とも評価指標としては WL_index を用いている．提案方式のカーブは greedy 方式と比べて常に原点側に位置する．このことは，遺伝的アルゴリズムの活用で，greedy 方式と比べ高性能なホワイトリストを作成できることを意味する．具体的には，同一の検知見逃しリスクに対する誤検知期間を $\theta$ =1 のとき 20%，$\theta$ =10（デフォルト値）のとき 15%削減できる．

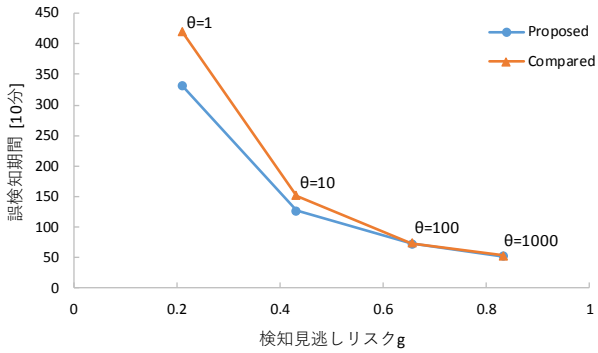


図 8　greedy 方式との比較
Figure 8　Comparison Greedy Approach.

図 9 に，d を 0 から 2 まで変化させた場合のトレードオフを示す．d を大きくするほど，検知見逃しリスクの低減を優先するホワイトリストが作成される．一方で，d を 0.5 より低くすると，提案手法では，検知見逃しリスク・誤検知期間が両方とも悪化する．このため，d は 0.5 以上の値を設定することが望ましい．

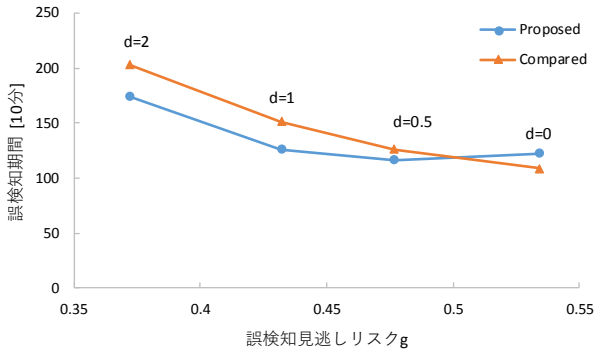


図 9　d の影響

Figure 9　Effect of d.

図 10 に，$TH_{max}$ を 4 から 100 まで変化させた場合のトレードオフを示す．$TH_{max}$ を大きくするほど，検知見逃しリスクの低減を優先する $W_h$ が作成される．一方，$TH_{max}$ を 4，すなわち TH と同値にした場合，検知見逃しリスク・誤検知期間が両方とも悪化する．このため，$TH_{max}$ は TH より大きい値を設定することが望ましい．

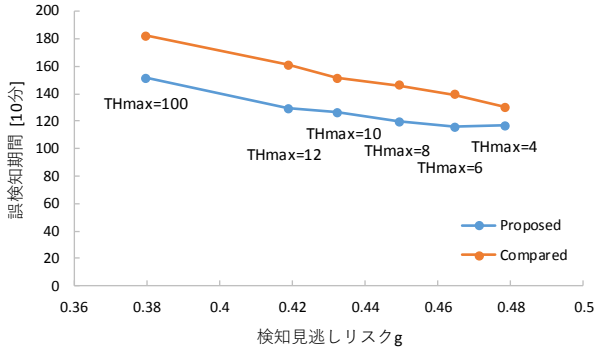


図 10　$TH_{max}$ の影響

Figure 10　Effect of $T_{max}$.

### (3)　リスク値の性能への影響

図 11 に R を 1 から 100 まで変化させた場合の，検知見逃しリスク $g(W_h,L_h,T)$と誤検知期間の関係を示す．R が大きいほど，時刻 T の直前で初めて記録された活動をホワイトリストに登録することのリスクは高く評価される．

図 11 からは，R が大きくなるほど，検知見逃しリスク g, 誤検知期間は共に改善するように見える. 前述のように，検知見逃しリスク g は，検知見逃しスコア $g_s(W_{full},L_h,T)$に対する $g_s(W_h,L_h,T)$の比率である．時刻 T までの全て活動を含む $g_s(W_{full},L_h,T)$は R に比例して大きくなる．一方，$g_s(W_h,L_h,T)$は $R^n$ $(n\leq 1)$に比例して大きくなる．このため$g \propto R^m(m \leq 0)$となり，$\lim_{R\to\infty} g = 0$が成り立つ．

一方，見逃しスコア $g_s$ を横軸とした場合の，誤検知期間との関係は図 12 の示すとおりになる．R が大きくなるほど，$g_s$ は大きくなり，誤検知期間は増加するというトレードオフの関係が現れる．

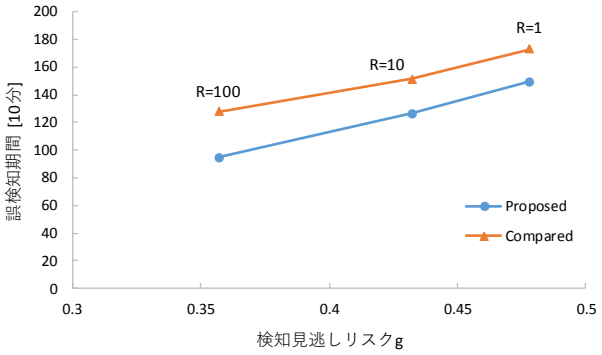


図 11　R が変化した場合の検知見逃しリスク g と誤検知期間の関係

Figure 11　Relation between false positive risk and false negative risk regarding R.

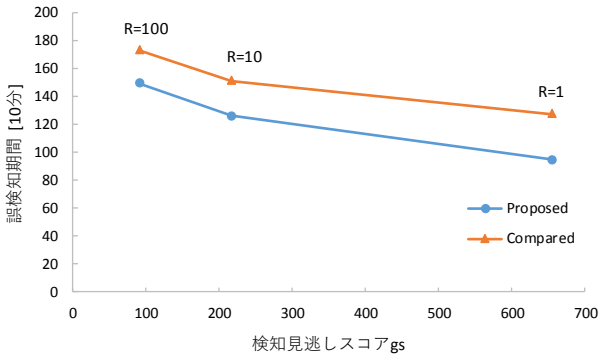


図 12　R が変化した場合の検知見逃しスコア $g_s$ と誤検知期間の関係

Figure 12　Relation between false positive risk and false negative score regarding R.

図 13 に，JPCERT/CC が公表している標的型サイバー攻撃に悪用されやすい Windows コマンド 18 種類を $B_{set}$ に登録した場合の性能評価を示す．提案方式では，図 7 の場合と同様に，$R^B$ の増加に伴い検知見逃しリスク及び誤検知期間は共に低減する．一方，greedy 方式では，誤検知期間が大幅に増加する．これは，発生頻度が多い Windows コマンドのリスクが増えるに従い，指定された検知見逃しリスク値を超えない範囲で，ホワイトリストに含めることができる活動数が減少するためと考えられる．

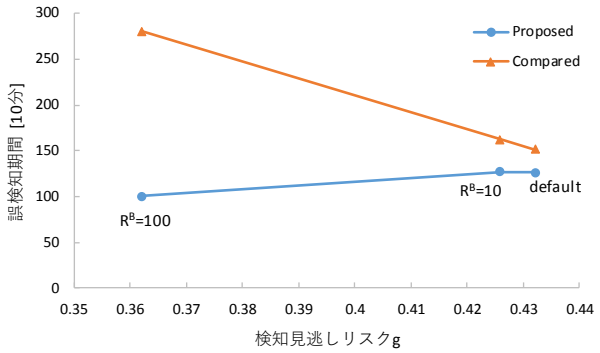


図 13　$R^B$ の影響

Figure 13　Effect of $R^B$.

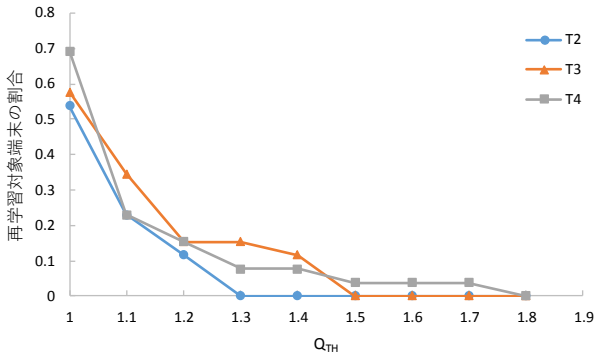


図 14　$Q_{TH}$ と再学習対象端末の割合

Figure 14　Fraction retraining hosts according to $Q_{TH}$.

### (4) ホワイトリストの経年変化と再学習

$T^k$ における，Q 値の平均・分散は表 4 の示すとおりになる．約一か月が経過しても，Q 値の平均はほぼ 1 のままである．このため，今回の実験で得られた範囲では，再学習が必要な端末は少数といえる．ただし，時間経過と共に標準偏差は増加傾向にあり，今後再学習が必要な端末は増加していくと考えられる．

図 14 に，$Q_{TH}$ と再学習対象端末の割合を示す．仮に，$Q_{TH}$=1.2, 即ち質が 20%以上悪化したホワイトリストを再学習対象とすると，T4 までに再学習対象は全体の 15%となる．たとえば，総端末数を 10,000 台とすると，総学習時間は約 36 時間となる．これは，業務が比較的少ない土日の 48 時間以内に再学習を完了できることを意味する．

$Q_{TH}$ の妥当性については今後長期的な実験により検証を進める必要がある．しかし，今回の実験で明らかになった範囲では，本報告で提案したホワイトリスト評価指標・作成方式・再学習手順は，再学習による演算負荷を低減し，多数の端末に対する効率的なホワイトリスト運用を実現する見込みがあるといえる．

表 4　Q 値の平均及び標準偏差

Table 4　Average and Standard Deviation of Q value.

| | $T^1$ | $T^2$ | $T^3$ | $T^4$ |
|---|---|---|---|---|
| 平均 | 1 | 1.020 | 1.058 | 1.044 |
| 標準偏差 | 0 | 0.142 | 0.213 | 0.239 |

### 5.3 考察

WL_index には 7 つのパラメータ，$\theta$，d，$TH_{max}$，R，$R^B$，window，TH が存在する．本節では，各パラメータの設定方法について検討する．

$\theta$ は，原則的には，誤検知と検知見逃しのどちらに重みを置くかにより決定される．考え方の 1 つとして，運用面などから誤検知期間に制約を設け，制約内で最適な $\theta$ を設定する，というものがある．たとえば，誤検知期間の比率を全端末平均 3%以内に抑えることを制約とした場合，図 7 より，本実験では $\theta$ =10 が望ましい値となる（実験期間 1 か月・720 時間の 3%は，約 1300 分であるため）．また，各端末の誤検知リスクスコアを一定値以下に抑えたい場合は，端末ごとに異なる $\theta$ を採用すればよい．

図 9 より，d は $\theta$ と比べると，値を変えた場合の誤検知期間の変動が小さい．特に 0<d≤1 の範囲では変動が殆どないため，検知見逃しリスクを最小化するという点で d=1 程度が望ましいと考えられる．

図 10 より，$TH_{max}$ も誤検知期間への大きな影響はない．このため，概ね，TH 以上の値を設定すれば良いと考える．

R の設定値は，ホワイトリスト作成 1 日前に観測された活動のリスクを，最小リスク(=1)の何倍程度と捉えるべきかに依存する．リスク評価手法である CVSS[13]ではリスクの最大値と最小値の差は 10 倍である．このため，デフォルトでは，R=10 程度とするのが妥当と考える．$R^B$ についても同様である．

TH 及び window は，拡散活動検知システム[5]の検知パラメータに依存する．デフォルトでは TH=4, window=1hr が望ましいと考える．

## 6. おわりに

本報告では，拡散活動検知システムで使用するプロファイル型ホワイトリストの性能向上を目的に，ホワイトリスト準最適化技術を検討・評価した．具体的には，IDS の評価指標である F-measure を参考に，誤検知及び検知見逃しリスクの観点から，ホワイトリストの質を定量評価する評

価指標を策定した．また，遺伝的アルゴリズムを用いて，与えられた活動ログを基に評価指標の点で準最適なホワイトリストを作成する方式を考案した．さらに，運用期間中にホワイトリストの再学習が必要なタイミングを判断する手順を検討した．

ある組織の 26 端末の活動ログを用い，本技術は誤検知・検知見逃しの両点で比較対象方式より 15%以上優れたホワイトリストを作成できることが明らかになった．また，本技術により，低負荷で，再学習タイミングを判断できることを確認すると共に，10,000 台の端末に対するホワイトリスト運用を実現できる見込みを示した．

今回の実験では，約 2 か月という短期間の観測結果を使用して評価を行っている．今後は，半年から一年以上の長期的な観測結果を用いて，提案指標及び提案方式の有用性を検証する必要がある．特に再学習手順に関しては，観測結果を基に適宜改善を行っていく．

本稿中で使われているシステム・製品名は，各社の商標または登録商標です．

## 参考文献


[1] IPA: 標的型攻撃/新しいタイプの攻撃の実態と対策, 入手先 < http://www.ipa.go.jp/files/000024542.pdf> (参照 2018-04).

[2] ラック: 「日本年金機構の情報漏えい事件から得られる教訓」公開のお知らせ, 入手先 <https://www.lac.co.jp/news/2015/06/09_news_01.html> (参照 2018-04).

[3] 中里ほか: ホスト型 IDS を用いた不審プロセスの特定, SCIS2015 予稿集 (2015).

[4] Keisuke Takemori, et al.: Detection of Bot Infected PC Using Destination-based IP Address and Domain Name Whitelists, IPSJ Journal, Vol.52, No.4, pp.1706-1716 (2011).

[5] 川口ほか: 不審活動の端末間伝播に着目した標的型攻撃検知, 情報処理学会論文誌, Vol.57, No.3, pp.1022-1039 (2016).

[6] A. Bifet, R. Gavaldà: Learning from time-changing data with adaptive windowing, Proc. 7th SIAM Int. Conf. Data Mining, pp.443-448 (2007).

[7] Eduardo Viegas, et al.: Stream Learning and Anomaly-based Intrusion Detection in the Adversarial Settings, IEEE ISCC2017 (2017).

[8] van Rijsbergen, C.J.: Information Retrieval. Butterworths, London (1979).

[9] JPCERT/CC: 攻撃者が悪用する Windows コマンド(2015-12-02), 入手先 < https://www.jpcert.or.jp/magazine/acreport-wincommand.html> (参照 2018-04)

[10] Verizon, 2017 DATA BREACH INVESTIVATION REPORT, available from <https://www.ictsecuritymagazine.com/wp-content/uploads/2017-Data-Breach-Investigations-Report.pdf> (2018-04 accessed).

[11] Verizon, 2015 DATA BREACH INVESTIVATION REPORT, available from <https://iapp.org/media/pdf/resource_center/Verizon_data-breach-investigation-report-2015.pdf> (2018-04 accessed).

[12] J.H. Holland: Adaptation in Natural and Artificial Systems, University of Michigan Press/MIT Press (1975).

[13] FIRST.org, Inc.: Common Vulnerability Scoring System SIG, available from < https://www.first.org/cvss/> (2018-04 accessed).