ユーザのコマンド履歴を用いたAdaboostによる認証手法改善の試み

中田 明秀 *　　小高 知宏 *　　白井 治彦 **　　黒岩 丈介 ***

Improvement of the Intrusion Detection Method Based on User Command History Using the Adaboost Machine Learning Meta-algorithm

Akihide NAKATA* and Tomohiro ODAKA*,
Haruhiko SHIRAI , Jousuke KUROIWA*****

(Received February 6, 2015)

In this study,we investigate the new intrusion detection method with Adaboost machine learning meta-algorithm. Adaboost can be used to improve their performance conjuncting multiple weak descriminators. We construct 6 weak descriminators for user command history and we conjunct these weak discriminator with Adaboost. As a result of experiment,we found that it is improved intrusion detections with Adaboost.

Key words **:** Intrusion Detection,User Certification,Command Chain,Discriminator,ROC Curve

1. 諸　言

私たちがコンピュータなどの端末を利用する際, ユーザ認証と呼ばれる認証システムを用いる. ユーザ認証[1][2]には, パスワード認証,IC カードなどの所有物を利用した認証, 生体認証といったものがある. これらの認証手法は, そのユーザが正当なユーザであるかどうかを確認するために用いられる. この認証手法を用いることで, 利用者が正しいユーザであるかを判断し, ユーザが安全に端末を使用できるようにするだけでなく, 他のユーザに不正に利用されるのを防ぐためのものとして用いられている.

しかし, ユーザ認証は, 一般的にはその端末の利用を開始する際にしか用いられていないという問題点がある. 例えば, パスワードであれば認証情報の解析や盗聴により, 必要となる情報が漏洩してしまう可能性が考えられる. こういった事態が起きてしまうと, 不正な利用者によって悪用され, 結果的に利用者が不当な被害を被ることになる恐れがある. また, 端末そのものが誤って不正な利用者を誤認してしまうといったことも考えられる. こういった場面を想定して, 再度正しいユーザかを確認するような認証手法が望ましいと考えられるが, 一度認証されたユーザについては再確認を求めるケースが少ないというのも現状である.

そこで, 本研究では従来の認証システムの問題点の解決策として, ユーザが入力したコマンド履歴の情報を認証手法に用いることにした. この手法では, まずユーザの特徴を学習するために, 事前に入力があったコマンド履歴の情報から, 各ユーザごとにモデルを構築する. 次に, 後から入力されたコマンド履歴の情報についても同様にして, モデルを構築する. この 2 つのモデルを比較した結果, どの程度類似しているかによって認証を行う. この認証手法であるが, モデルの比較の仕方は千差万別であり, また最適と思われる解析手法があまりないこと

*大学院工学研究科 原子力・エネルギー安全工学専攻
**工学部技術部
*** 大学院工学研究科 知能システム工学専攻
*Nuclear Power and Energy Safety Engineering Course, Graduate School of Engineering
**Dept. of Technical, Dept. Engineering
***Human and Artificial Intelligent Systems Course, Graduate School of Engineering

から, 一つの手法だけを用いた認証手法では, あまり信頼できる結果にならないと考える.

そこで, 複数の手法を組み合わせることで, より正確な認証結果を得るための方法として Adaboost[3][4][5] という機械学習を用いることにした. これにより, それぞれの手法と Adaboost により学習した結果とを比較して, どの程度, 認証精度が上がるのかをみることにした. その結果,Adaboost を用いた方が手法を組み合わせる前よりも正しいユーザか侵入者ユーザかをより正しく認識できるようになった. また, 本研究で行った手法全体についても評価したところ, 誤認識率を下げることにつながるという結果になった.

以下, 本論文の流れは次のとおりである. 第 2 章では, コマンド履歴を用いた認証の具体的な流れについて紹介し, そこで必要となる解析手法についての内容及び,Adaboost そのものについて説明する. 第 3 章で, 本研究で用いたコマンドデータである Schonlau データの特徴及び, コマンド履歴の解析方法について示す. 第 4 章で本研究の実験結果により, 各手法及び Adaboost による結果と全体としての認証結果について触れる. そして, 第 5 章で考察を行い, 第 6 章で本論文のまとめを行う.

2. コマンド履歴を用いた認証手法と解析手法について

2.1 コマンド履歴を用いた認証手法

ここで, 本研究で用いたコマンド履歴による認証手法の仕組みについて示す. コマンド履歴の情報を認証手法として用いるためには, そのユーザの情報を解析することによって特徴化することが必要になる. そこで本研究では, 図 1 のようにして, モデルを構築することで比較を行った.

まず, 学習モデルの構築として, 正当なユーザが入力したすべてのコマンド履歴の情報を用いて学習を行う. ここでは, ユーザが入力したコマンドの種類や出現頻度, 前後関係といったものを学習させる. これを学習モデルと定義し, のちに構築された検査モデルとの比較に用いる.

次に, 学習モデルによる十分な学習が終わったら, 検査モデルの構築に入る. こちらも先ほどと同様にして, ユーザが入力したコマンド履歴の特徴を元にモデルの構築を行うが, 先ほどとは異なり, 正当なユーザによるものだけではなく, 侵入者ユーザのモデルも構築する. ここで構築したものを検査モデルという形で表す.

ここで, 学習モデルと検査モデルを元に, そのユーザが同一のユーザであるかどうかの比較を行う. 比較には,2.2 で紹介する判別手法に基づいて評価を行い, その

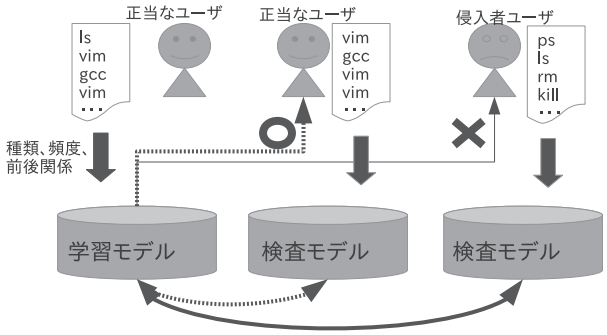

図 1: コマンド履歴による認証手法の仕組み

差異がどの程度のものであったかによって正当なユーザか侵入者ユーザかどうかを判断する. 一般的には, 正当なユーザであれば, 学習モデルに近い検査モデルが構築されるため, 正しいユーザであると認証されやすくなる. 一方で, 侵入者ユーザであれば, 学習モデルとは異なる検査モデルが構築されることで, 侵入者ユーザであるとみなし拒否されやすくなる. このような形でコマンド履歴を用いた認証を行う.

コマンド履歴は, その情報の解析の仕方によって, 様々な見方が出来ることから, 多くの判別手法を考えることができるといった利点がある. しかし, それぞれの判別手法には利点や欠点があるため,1 つの手法だけではあまり正確な結果が出ないと思われる. そこで, 本研究では Adaboost という手法を用いることにより, 複数の手法を組み合わせることで総合的な認証評価を行うことにした.

Adaboost は機械学習アルゴリズムの一種で, 与えられたいくつかの学習アルゴリズムを識別器として, 複数の識別器と組み合わせて用いることで, その手法の精度を改善することができる. このアルゴリズムは, それぞれの識別器が正解したかどうかを判断し, 最終的には重みという形で評価する. 具体的には識別器が正しく判断したものは重みが小さく, 間違って判断したものは重みが大きくなるように更新する. この重みの違いを利用して, それぞれのデータごとに正しい選択がされるような強識別器を構築して, 最終的な認証評価を行う.

本研究で用いる場合, コマンド履歴のデータを用いた比較手法を識別器として比較に用いることにした. Adaboost に基づいて複数の識別器を組み合わせることでより強力な識別器を構築し, 認証の精度を改善することを本研究の目的とした.

2.2 Adaboost に用いた識別器について

2.1 では本研究で用いた学習アルゴリズムとして Adaboost を取り上げたが, この手法を用いるためには識別

器を複数用意する必要がある. この識別器の一つ一つはあまり精度の良いものではないため, これらの識別器を弱識別器と呼ぶ. ここで, 弱識別器を構成するために用いた解析手法として, 主にコマンドのヒット率,COS 類似度, 及び TF-IDF の 3 つの手法を取り上げた.

2.2.1 コマンドのヒット率

コマンドのヒット率では, コマンド履歴に基づいて構築された 2 つのモデルを比較し, 一致したコマンドの割合を求める. これは, 検査モデルに現れたコマンドが学習モデルにどれだけ含まれていたかで評価を行う手法である. コマンド履歴を用いて構築されたモデルについて, 学習モデル, 検査モデルに含まれるコマンドの総数をそれぞれ $n(X)$,$n(Y)$ とする. このとき,2 つのモデルに共通して現れるコマンドの総数は $n(X \cap Y)$ であるから, コマンドのヒット率は $P(X \mid Y)$ となり, 式 (1) に基づいて求めることができる.

$$P(X \mid Y) = \frac{n(X \cap Y)}{n(Y)} \tag{1}$$

本研究で用いたコマンド履歴については 3 章で紹介するが, この情報を用いて構築される学習モデルには各ユーザごとに比較的出現しやすいコマンドがあることが分かっている. また, 入力コマンドの性質上, 出現頻度の高いコマンドであっても, 一部の区間で大量にそのコマンドが現れることは少なく, ある程度一定の頻度で現れやすい. そのため, 学習モデルに比べてデータ量の少ない検査モデルであっても, 学習モデルで出現頻度の高かったコマンドが現れやすいと考えられるため, 検査モデルと学習モデルが同一のユーザのものであれば, 高い割合を示す傾向がある.

2.2.2 COS 類似度

COS 類似度 [8] は, 文書中に現れた単語を用いて 2 つの文書の類似度を計る尺度としてよく用いられる手法である. ここでは, 学習モデルに現れた各コマンドが, 検査モデルにおいても同頻度あるいはそれに近い頻度で現れるであろうということに着目した. 本研究ではユーザが入力したコマンド履歴中に含まれるコマンドを単語の代わりとして比較に用いることにした. コマンドは, 文書に例えると, 本文中に含まれる単語に相当すると考えられることから, この手法を用いることが出来ると考える.

学習モデルに含まれる各コマンドの要素, 検査モデルに含まれるコマンドの要素をそれぞれ $\overrightarrow{x}$,$\overrightarrow{y}$ とし, 各コマンドの出現頻度を x_a,y_a と表した時の COS 類似度は, 式 (2) のようにして表される. この値が 1 に近いほど類似性が高く,0 に近いほど類似性が低いことを意味しており, 類似性が高いほど同一のユーザである可能性が高いと考えられる. これを利用して正当なユーザか別のユーザかを判別することができると考えられる.

$$\cos(\overrightarrow{x}, \overrightarrow{y}) = \frac{\overrightarrow{x} \cdot \overrightarrow{y}}{|\overrightarrow{x}||\overrightarrow{y}|} = \frac{\sum_{a=1} x_a y_a}{\sqrt{\sum_{a=1} x_a^2}\sqrt{\sum_{a=1} y_a^2}} \tag{2}$$

2.2.3 TF-IDF

TF-IDF[9] は情報探索でよく利用される重み付け手法の一つである. この手法はいくつかの文書が与えられた時にそれぞれの文書を特徴づけている単語を高く評価するのに用いられる手法である.

TF-IDF は出現頻度 tf(term frequency) と逆文書頻度 idf(inverse document frequency) の積で式 (3) のように表される. tf とは, その文書 b におけるコマンド n_a の出現回数を, その文書中の単語の総数で割ったものである. 一方,idf は,$|D|$ は文書の総数,$|\{d : d \exists n_a\}|$ はコマンド n_a を含んでいるユーザ数を表しており, コマンド n_a の希少性を表している.

$$tf_{a,b} \cdot idf_a = \frac{n_{a,b}}{\sum_c n_{c,b}} \times \ln \frac{|D|}{|\{d : d \exists n_a\}|} \tag{3}$$

これはあるコマンド n_a が文書に現れる頻度 (文書頻度)df の逆数をとっており, 他の文書には現れにくいような単語は高く評価される. 一方で, 一般的に現れる単語については低く評価される. そのため,TF-IDF は他の文書には現れにくい単語でかつ, その文書中における出現頻度が高いとよいとされる.

コマンド履歴についても, 同様な性質があることが考えたため, 本研究では, 文書をコマンド履歴, 単語をコマンド履歴中のコマンドに置き換えてこの手法についても用いることにした. しかし,TF-IDF は文書の各単語について重みづけを行うものであり, この手法を用いただけでは, ユーザの判別はできない. そこで本研究では,TF-IDF により各単語について重み付けを行った後,COS 類似度と合わせて用いて, ユーザの判別を行うことにした.

2.3 Adaboost

Adaboost は弱識別器の結果を組み合わせることで, 強識別器を生成することができるということを 2.1 で述べた. ここでは,Adaboost により, どのようにして学習が行われ, 強識別器が生成されるのかについて示す.

まず,Adaboost を用いる前に, なんらかの手法により良し悪しを判別出来る弱識別器を M 個用意する. この弱識別器について, 正解したかどうかを値として返す

ようにし, 正解であれば 1, 誤りであれば-1 を返す関数を $f_h(i) = \{1, -1\}$ のように表すことにする. これを与えられた I 個のデータについて, 以下のようにして Adaboost により学習を行う. 重みの初期値として, 比較を行うデータについて同じ重みを式 (4) のようにして与える.

$$D_1(i) = \frac{1}{I}, i = 1, \ldots, I \tag{4}$$

重みの値は弱識別器が選択される度に重みの更新が行われ,T 回の学習を経て, 誤って判断されやすいデータを検出し, 正しい選択がされるように学習が行われる. 次に,$t(1 \leq t \leq T)$ 回めの学習として, 用意した M 個の弱識別器から, 最もエラーが少ない弱識別器 h_t を選択する. このとき, エラー率 ε_t は, 誤りと判断されたデータの重みの総和であり, 式 (5) のようにして求めることが出来る.

$$\varepsilon_t = \min\{\sum_{i=1}^{I} D_t(i)\{f_{h_t}(i) = -1\}\}, 1 \leq h_t \leq M \tag{5}$$

次に, 選択された弱識別器を用いて, その識別器がどれだけ正しく判断されたかを信頼率 α_t と言う形で表す. この値は, 先ほど求めた式 (5) を用いて,α_t を式 (6) のようにして求められる. ε_t であるが, この値が 0.5 を越えてしまうと,α_t の値が負の値となってしまい, 正しい学習が出来なくことに気をつける.

$$\alpha_t = \frac{1}{2} \ln \frac{1 - \varepsilon_t}{\varepsilon_t} \tag{6}$$

ここで,α_t の値を用いて,t 回目の重みの更新が行われる. 式 (6) で求めた α_t の値を用いることにより, 重みを更新した値 d_{t+1} が式 (7) により求められる. この式から, 正しいと判断されたデータについては $\{-\alpha_t \cdot f_{h_t}(i)\}$ の値が負になるため, 重みの更新が行われる前よりも重みの値が小さくなる. 一方で, 誤りであると判断されたデータについては重みが大きくなるように更新が変化する.

$$d_{t+1}(i) = D_t(i)\exp\{-\alpha_t \cdot f_{h_t}(i)\} \tag{7}$$

しかし, 式 (7) による重みの更新を行った場合, 重みの合計した値が 1 にならなくなってしまう. そこで, 式 (8) により, 重みの総和が 1 になるように正規化が行われる. これにより,t 回めの重みの更新が完了する.

$$D_{t+1}(i) = \frac{d_{t+1}(i)}{\sum d_{t+1}(i)} \tag{8}$$

これを式 (5)〜(8) により T 回学習を繰り返すことで, 誤りのデータの重みの比重が大きくなり, 精度の良い弱識別器が選ばれやすくなるように学習が行われる.

最後に, すべての弱識別器の結果を考慮した強識別器を生成する. この強識別器は, 比較に用いたデータと各学習により選択された弱識別器の信頼率を掛け合わせたものの総和が 0 を超えたかどうかにより,1 または-1 を返したものである. そのため, この値が正か負かどうかで強識別器による最終的な判別が行われ, 式 (9) のように表される. 式 (9) は, 最も精度が良いとされた弱識別器の結果を掛け合わせたものであるため, 個々の弱識別器よりも精度の高い真偽結果が得られると考えられるのである. 上記の内容をまとめると, 図 2 のようなアルゴリズムで表される.

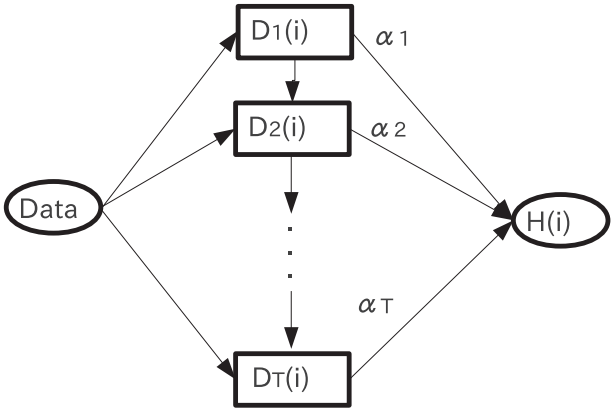

図 2: Adaboost による重みの更新の流れ

$$H(i) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t f_{h_t}(i)\right) \tag{9}$$

3. 実験方法

3.1 実験に用いたコマンド履歴について

コマンド履歴を用いて認証を行うためには, ユーザの学習モデルと検査モデルを構築し, 比較を行う必要があることを第 2 章で述べた. ここで,2 つのモデルを構築するためには, 複数のユーザからデータの収集を行うだけでなく, 大量のコマンドデータを扱った長期的なデータが必要である.

しかし, ユーザによってはコマンドの種類や入力の頻度, コンピュータの利用状況等も異なるため, すべてのコマンドデータを収集したとしても十分な実験結果が得られない可能性も考えられる. そう考えると, 本研究での良し悪しを評価するためは, 他の研究でも用いられているようなコマンドデータを用いることが望ましいといえる.

そこで, 本研究では共通のコマンドデータとして, 一般的に公開されている Matthias Schonlau[6][7] が紹介しているコマンドデータを用いた. このデータは 50 人分のコマンドデータについて同じ量のコマンドデータが用意されており, モデルを構築した際のデータの情報量

の差が少なく, 研究を行うのに適したデータであると思われる. 以下, このデータを Schonlau データと呼ぶことにして, 本研究で取り扱うこととした.

3.2 Schonlau データの特徴

公開用のコマンドデータの例として, 本研究では Schonlau データを取り上げたが, 一般的に次のような特徴があることが分かっている. Schonlau データは 50 人のユーザについて各 15000 個のコマンドが用意されている. このうち, 初めの 5000 個のコマンドデータが学習用のデータであり, 残りの 10000 個のコマンドデータが比較用のデータとなっている. 本研究では, 前者を学習データ, 後者を検査データと呼ぶことにする.

このコマンドデータでは正当なユーザが入力したコマンドのみで構成されたコマンドデータとなっている. 学習データについては正当なユーザのモデルを構築するためのコマンドデータであるため, 侵入者ユーザが入力したコマンドが入力されていることはない.

一方, 検査データでは, 入力されたコマンドデータが正当なユーザのものかどうかをテストするためのデータである. このコマンドデータについては,100 個のコマンドデータを 1 セッションとした基本単位で構成されており, 合計 100 のセッションで成り立っている. 1 セッションとは, そのユーザが端末にログインしてからログアウトするまでに入力されるコマンドの量とここでは考えることにする. 1 つ 1 つのセッションに含まれるコマンドデータをセッションデータと呼ぶことにする.

このセッションデータであるが, セッションの途中でユーザが変わるということはなく, 基本的には正当なユーザのセッションで構成されている. しかし, 一部のセッションにおいて正当なユーザのものとは異なるセッションデータに入れ替えられている可能性があり, そのセッションに関しては連続して現れやすいといった性質もある.

本来は, 侵入者ユーザが正当なユーザになりすまして入力するため, そのコマンド履歴はもっと悪質なものである. そのため, 別のユーザのものと入れ替えたデータでは, 本当の意味での侵入者のユーザのセッションデータとは趣旨が異なる. しかし, 別のユーザのセッションデータと入れ替わっているだけであっても, 正当なユーザかどうかを区別するという点においては何ら変わりはないはずである. そこで, 本研究では入れ替えられたセッションを侵入者セッションであると見なして実験を行うことにした. 上記の内容をまとめると, 表 1 のようにして Schonlau データが構成されていることが分かる.

表 1: Schonlau データの構成

ユーザ数	50 人	
名称	学習データ	検査データ
コマンド数	5000	10000
セッション	なし	あり
セッション数	—	100
特徴	すべて同じユーザのコマンドデータ	侵入者のセッションを一部含むコマンドデータ

表 2: 弱識別器の 6 つの手法の内訳

method1	1 コマンドによるコマンドのヒット率
method2	1 コマンドによる COS 類似度
method3	1 コマンドによる TF-IDF
method4	2 コマンドの連鎖によるコマンドのヒット率
method5	2 コマンドの連鎖による COS 類似度
method6	2 コマンドの連鎖による TF-IDF

3.3 コマンド履歴を用いた比較方法について

次に,Schonlau データを用いて, 学習データのコマンド履歴とセッションデータのコマンド履歴を比較する方法について示す. Schonlau データを用いての学習データとセッションデータの比較方法であるが, 本研究では 1 コマンドによる場合と連続した 2 コマンドの連鎖による場合の 2 つのパターンを考慮することにした. これは,1 つ 1 つのコマンドを見て比較していくよりも, コマンドの前後関係を考慮した方が, より細かいユーザの特徴を検出でき, 結果が良くなるのではないかと考えたからである. また, 解析手法については第 2 章で述べた, コマンドのヒット率,COS 類似度,TF-IDF の 3 つの手法について行っており, これらの組み合わせからなる計 6 つの手法を Adaboost に用いる弱識別器として扱うことにした (表 2).

ここでは, あるユーザの学習データのコマンドデータが図 3 のように構成されていた場合について COS 類似度を用いて比較を行う場合の例について示す. このコマンドデータは入力されたコマンドの順に記録されているものであるが, 同じ種類のコマンドが複数回に渡って入力されている場合もある. そこで, 学習モデルの情報量をなるべく少なくするために, 複数回表れたコマンドは 1 回だけ表記するようにし, 図 3 のように書き換えることにする. これを出現頻度表と呼ぶことにする.

次に, 先ほど構築した学習モデルを用いて比較を行

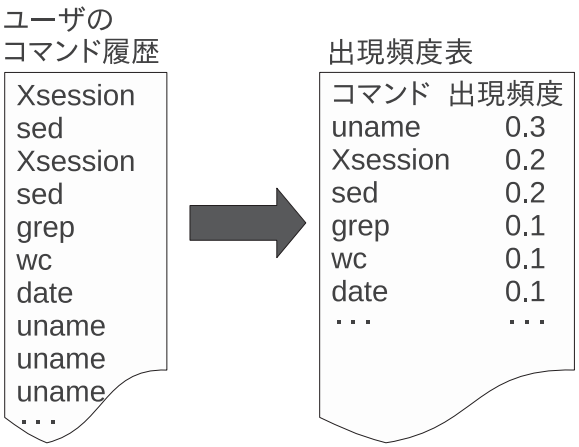

図 3: コマンド履歴による 1 コマンドの出現頻度表の生成

コマンド	学習モデル 出現頻度	検査モデル 出現頻度
uname	0.3	0.0
Xsession	0.2	0.3
sed	0.2	0.2
grep	0.1	0.1
wc	0.1	0.1
date	0.1	0.1
Fvwm	0.0	0.1
xdm	0.0	0.1
・・・	・・・	・・・

図 4: 学習モデルと検査モデルとの比較

う場合について考える. 検査モデルの出現頻度表についても, 同様にして出現頻度表を作成する. 例えば, 学習モデルと検査モデルの 2 つのモデルを用いて COS 類似度を求める場合, 次のようにして行う. ここでは,2 つの出現頻度表をまとめたものを図 4 のようにして表し, 比較に用いた. この場合, 学習モデルと検査モデルに共通して現れたコマンドは Xsession,sed,grep などであり, 片方のモデルに現れなかったコマンドとして uname や fvwm などがあったとする. ここで,2.2 の式 2 により, 学習モデルに現れたコマンドを $\vec{x}$, 検査モデルに現れたコマンドを $\vec{y}$ とすると, 類似度は約 0.685 となる. この値が各手法ごとに任意に設定した閾値により, 閾値を上回る類似度の値が得られれば正しいユーザとして判断されるということになる.

コマンド連鎖による解析手法についても, 先ほどと同様にして解析を行うことが出来る. コマンド連鎖の場合, 図 5 のように前後の 2 つのコマンドを組にして構成される. この例で入力されたコマンドは 10 個である

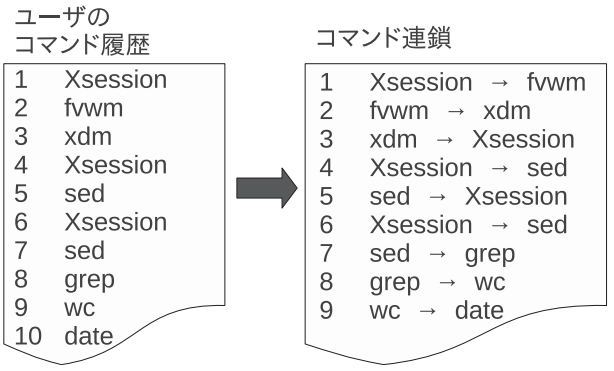

図 5: コマンド履歴によるコマンド連鎖の組の生成

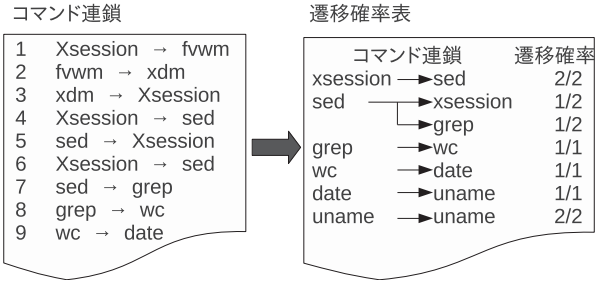

図 6: コマンド履歴によるコマンド連鎖の出現頻度表の生成

が, コマンド連鎖の場合は必ず元のコマンドの数よりも 1 つ少なくなることに注意する.

そして, コマンドの連鎖の場合においても同様にして, 図 6 のようにして書き換えを行ったコマンド連鎖の遷移確率表を生成し, 比較に用いた. これを学習モデル, 検査モデルともに構築し, 比較を行った.

4. 実験結果

4.1 各弱識別器による結果

本研究の実験結果として, まず 6 つの弱識別器による結果について示す. その結果, 大きく分けて 2 つのパターンの結果が見られた. 図 7 や図 8 は対象となったユーザについての結果を示しており, 横軸は検査データの 100 つのセッション, 縦軸はそのセッションにおいて,3.3 の表 2 のそれぞれの手法による結果について示す.

一つは, 図 7 のように 6 つの手法すべてにおいて, 一部のセッションで値が急激に低くなるという顕著な結果が得られた. このグラフからセッション 28〜40 にかけて 0 に近い値をとっており, どの手法においてもこの区間において侵入者ユーザのセッションであると判別しているという結果になり, 手法を組み合わせなくても検出出来る例があることが分かった.

一方で, 多くの実験結果では図 8 のような結果になり,

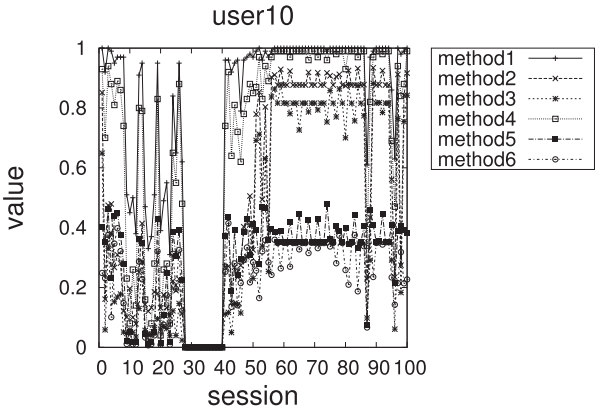

図 7: 各弱識別器による値の推移の例 1

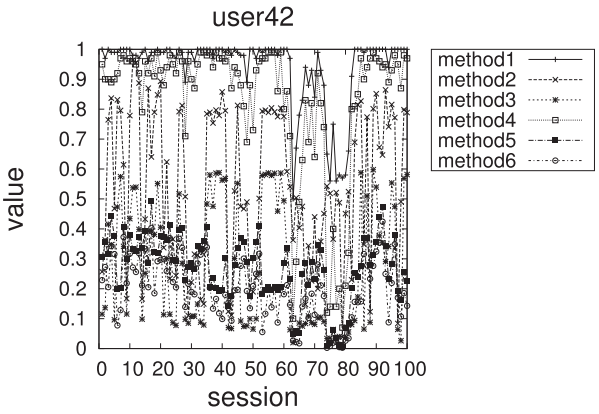

図 8: 各弱識別器による値の推移の例 2

すべての手法で急激に値が低くなるような例はあまり見られなかった. また, 各手法ごとのグラフの変位の仕方は似ているものの, 手法ごとにとりうる値が大きく異なるという結果になった. method1 や method4 のようなコマンドのヒット率を示す場合についてはやや高い値を示しているが,COS 類似度や TF-IDF といったコマンドの性質を見ていくような手法ではやや低い値をとっていた. このことから,6 つの手法を同じ基準で比較を行うことはできないと考えから, 各手法ごとのグラフの変位の傾向から適した閾値を設定した上で真偽の結果を決定した.

4.2 Adaboost による結果

次に,Adaboost により 6 つの手法を組み合わせた結果の例を示す. 図 9 の横軸はセッション, 縦軸はそのセッションに対する $H(i)$ の値を示しており, 図 8 と同じユーザについて, それぞれの手法ごとに閾値を設定して重みの更新を行った結果を示している. ここで式 (9) により, 正の値をとったセッションは正当なユーザのセッションであると判断し, 逆に負の値をとったセッションについては侵入者のセッションであることを示している. そ

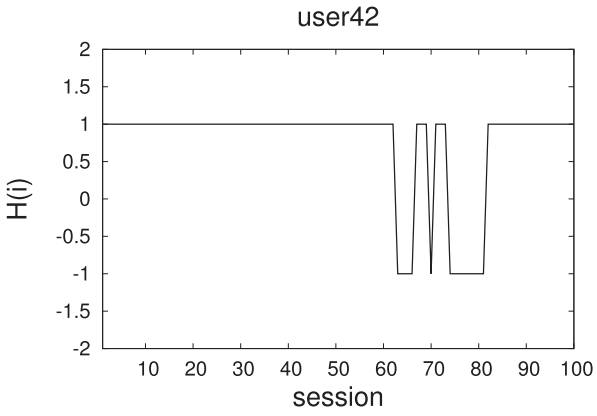

図 9: Adaboost の強識別器における値の推移の例

の結果, セッション 60〜80 にかけて,$H(i)$ の値が負の区間が見られ, この区間のセッションが侵入者のセッションである判断することが出来る. 実際に,Schonlau が公開しているデータ [6] に含まれている侵入者のセッションの分布表と照らし合わせてみると, セッション 60〜80 が侵入者のセッションとなっている. この結果から侵入者のセッションを検知することが出来ていることが分かった. しかし, 誤って正当なユーザのセッションで負の値を示したり, 侵入者のセッションで正の値を示したりするケースも見られた.

4.3 各弱識別器と強識別器との比較による評価

4.2 の結果から,Adaboost を用いることで, 侵入者のセッションを検出することができたが, 手法を組み合わせることで認証の精度が上がったのかについての検証も行った. 表 3, 図 10 は, 本研究で行った 6 つの手法と Adaboost についての誤認識率を FAR と FRR の関係を用いて示したものである. FAR は他人受入率であり, 誤って侵入者のユーザを通してしまった割合を示したものである. 一方で,FRR は本人拒否率であり, 正当なユーザを拒否してしまった割合を示す. そのため, どちらも可能な限り, 低く抑えることが重要である. しかし, 実際には,FAR を低くすると,FRR は高くなってしまい, 逆に FRR を低くすると,FRR が高くなるという関係になっている.

そこで, 表 3 は,FAR が同程度になるように換算したときの FRR について示し, 認証に失敗したときの全体のリスクの評価について示したものである. 4.1 の結果でも示したとおり, 手法ごとにとりうる値が大きく異なったために FAR を調整するのが難しく,15.06%〜25.21%と 10%近く値が開いている.

一方で,FRR のほうでは明確な結果が出ており,FAR で高い値を示した method6 については,FRR の値も同様

表 3: 各手法及び Adaboost による FAR と FRR との関係

	FAR	FRR
method1	21.58%	13.88%
method2	17.98%	25.95%
method3	17.04%	15.53%
method4	16.69%	17.47%
method5	22.33%	32.04%
method6	25.21%	38.21%
Adaboost	15.06%	6.35%

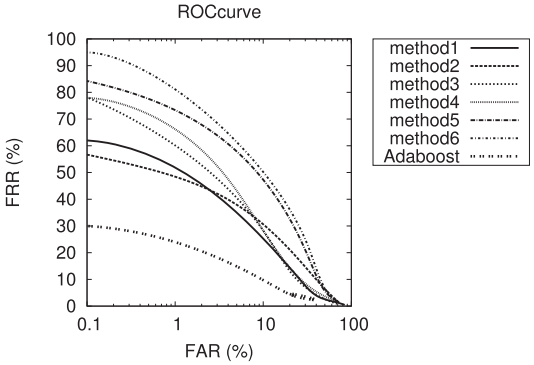

図 10: ROC 曲線による誤検知率の評価

に高くなってしまったため, この 2 つの手法についてはあまりよい結果ではなかったと考えられる. それ以外の method の FRR についてみていくと,13.88%～38.21%とかなり開きがあったが,Adaboost ではそれよりもさらに低い値である 6.35%という結果が得られた. この結果から,FAR,FAR の両方に関しても, どの 6 つの手法よりも低い値を示した Adaboost が最も認証の精度が高いという結果になり, 手法を組み合わせる前よりも精度が上がっていることが見られた.

次に, 図 10 は本研究で行った 6 つの手法及び Adaboost の FAR と FRR の関係を示した ROC 曲線である. 図 10 では, 横軸に FAR をとり, 縦軸はその FAR に対する FRR を示したものである. FAR については侵入者の誤検知率であり, 可能な限り 0%に近づけることはできるが,0%にすることはできないため, 対数をとったものである.

図 10 から,Adaboost の曲線は FAR が 20%未満の値に関して,FRR の値が他のどの手法よりも値が低くなっていることが分かる. また, 曲線の変曲点も, 他の 6 つの手法よりも左下側にあると見られることから,Adaboost の方が FAR,FRR ともによい結果になっていると考えられる. このグラフから, 最適と思われる FAR,FRR に注目した場合, ともに 10%近くに抑えられていることが分かる. 本研究で行った他の 6 つの手法とを比べると, 誤検知率が低いことから, かなり精度の良い結果が得られたのではないかと考えられる. これを一般的な認証手法と直列的に用いた場合, 誤検知率も 10 分の 1 程度に抑えられることになる. このことから, 従来の認証手法と併用することを考えれば, 十分な結果であると思われる.

一方で, 従来のバイオメトリクス認証手法とを比較した場合, 本研究で得られた FAR,FRR の値は, まだまだ充分ではないと考えられる. 例えば, 指紋認証の場合,FAR は 0.001%,FRR は 0.5%であると言われている.[10] 本研究では, ログイン中のユーザを対象としたものであり, 一般的なものとは異なる認証手法ではあるが, まだまだ改善できる可能性もあると考えられる.

5. 考察

本研究では,Schonlau データのコマンド履歴を用いて, 学習モデルの構築と各セッションの検査モデルの構築を行い, これらが同一のユーザに対して似た特徴を示しているのかどうかについての比較を行った. 比較を行う際, 一つの手法だけでは信頼性のある結果が得られないと考え,Adaboost を用いて複数の手法を組み合わせることで認証の精度が上がるのではないかと考えた. そこで,4.1 の解析結果により, まず 6 つの手法において学習モデルと検査モデルを比較し, 各セッションでどれぐらいの値をとるのかについて示した.

その結果, 図 7 のような結果が多ければ, どの手法においても侵入者のセッションを検出できているため,Adaboost を用いる必要性がなくなる可能性も考えられた. しかし, 多くの場合, 図 8 のようになり, それぞれの手法で閾値を個別に設定して, 正当なユーザのセッションかどうかを判別することになるという結果となった. このことから, 一つの手法だけでは正当なユーザのセッション及び侵入者ユーザのセッションを判別することは難しく, 複数の手法を組み合わせる必要性があることが分かった.

次に, 実際に Adaboost を用いて手法を組み合わせることによって強識別器を生成したところ, 図 9 のような結果になり, 侵入者のデータが含まれている箇所においては,$H(i)$ の値が 0 を下回っており, 侵入者のセッションを検出できることが確認できた. また, 表 3 の結果から,6 つの手法が同程度の FAR において,Adaboost では FRR が最も低くなるという結果が見られた. これは, それぞれの手法で正当なユーザや侵入者ユーザを検出しやすいパターンがあるからではないかと考えられる. そのため, 一つの手法だけでは, うまく特徴を抽出することができず, 誤って正当なユーザを拒否したり, 侵入者

ユーザを受け入れたりしてしまうということが考えられる.

これが Adaboost により, 手法を組み合わせて総合的に評価することによって, 誤って認識される確率を下げることにつながるという結果になった. このことから, 手法を組み合わせる前よりも侵入者のセッションを検出しやすくなったのではないかと考えられる. よって, コマンド履歴を用いた認証手法においても, 複数の手法を組み合わせることで精度を上げるという考え方は有用であると考えられる.

その一方で複数の手法を組み合わせても, それぞれの手法で誤って正当なユーザ及び侵入者のユーザと判断してしまっている場合については, 本手法ではそのまま誤って検出されてしまうことが分かった. この問題に関しては, それぞれの手法の精度があまり良くないものが多かったり, 似たような手法の結果を多く組み合わせたりしたものによると思われる. これらの手法をある程度重ねて組み合わせることは, データの信頼性を上げるのに役立つと考えられるが, その中にはいずれの手法においても間違って判断してしまったものも含まれていた. そのような結果が多く現れてしまうと,Adaboost を用いてもあまり改善されなくなるという結果になってしまうため, 明確な特徴のある手法を組み合わせることも必要であったと思われる.

6. 結　言

本研究では, 従来の認証方法では解決できない問題点の一つとして, ログイン後にユーザの認証を行うために, コマンド履歴を用いた認証手法を取り上げた. コマンド履歴を用いた認証手法では, 正当なユーザであるかを明確に判断する基準がないため, 研究で用いた複数の手法を組み合わせることで信頼性のある結果にするために機械学習の一つである Adaboost に注目した. ここで用いた手法は, 基礎的な解析方法ではあるが, 出現したコマンドのヒット率や COS 類似度,TF-IDF について,1 つ 1 つのコマンドを対象にするのか,2 つのコマンドの連鎖を対象にするのかを手法として用いた.

その結果, 一つの手法だけでは曖昧な結果となったが, 複数の手法を用いることでより侵入者のセッションを正しく判別することが出き,FAR が 15%台に対して,FRR は 6%という結果になり, 手法を組み合わせる前よりも改善が見られた. これにより, コマンド履歴を用いた認証手法においても手法を組み合わせることが有用であるという結果が得られた. コマンド履歴を用いた認証手法には, 従来の認証手法にはない特徴を持っていることから, それを考慮して本手法とを組み合わせることが出来れば, 実際の認証手法にも役立てることが出来るのではないかと考えられる.

参考文献

[1] 高田　哲司:セキュリティとユーザビリティ特集 個人認証におけるセキュリティとユーザビリティ, ヒューマンインタフェース学会誌,Vol9,No.1(2007).

[2] 吉田　隆:高精度化する個人認証技術ー身体的、行動的認証からシステム開発、事例、国際標準化まで, 美研プリンティング株式会社,215-223(2014).

[3] 塚本　浩司, 颯々野　学:AdaBoost と能動学習を用いたテキスト分類, 自然言語処理,146-13,81-88(2001).

[4] 竹之内　高志, 金森　敬文, 村田　昇, 江口　真透:ブースティングとそのロバスト化, 数理解析研究所講究録,1439,111-127(2005).

[5] 金森　敬文, 畑埜　晃平, 渡辺　治:ブースティングー学習アルゴリズムの設計技法ー, 森北出版株式会社 (2006).

[6] Matthias Schonlau,William DuMouchel,Wen-Hua Ju,Alan F.Karr,Martin Theus and Yehuda Vardi:Computer Intrusion:Detecting Masquerades,Statistical Science 2001,Vol.16,No.1,1-17(2001).

[7] Matthias Schonlau,Martin Theus:Detecting masquerades in intrusion detection based on unpopular commands,Information Processing Letters 76(2000),33-38(2000).

[8] 中谷　圭吾, 鈴木　優, 川越　恭二:文書間類似度とキーワードを用いた Web リンク自動生成手法,DBSJ Letters,Vol4,No.1(2005).

[9] 北村　順平, 青野　雅樹:ウェブサイト間の類似度を用いたウェブスパムの検出,DBSJ Journal,Vol8,No.1, 日本データベース学会論文誌 (2009).

[10] @IT:導入前に知っておきたいバイオメトリクス認証 (前編):http://www.atmarkit.co.jp/fsecurity/special/44biomet/biometrics01.html,site accessed at Nov 22,2003.