コネクションベース方式による踏み台攻撃検出手法の提案

竹尾大輔[†] 伊藤将志[†] 鈴木秀和[†]
岡崎直宣[††] 渡邊晃[†]

攻撃者が不正アクセスを行う場合，攻撃者の身元を隠すために複数の踏み台ホストを経由する踏み台攻撃を行っている場合が多い．このような攻撃を検出する方式として，これまで踏み台ホストをはさんだリモートログインストロークの時間的な相関関係を見るという方法があった．しかし，この方法では踏み台ホストへのアクセスと踏み台ホストから被害ホストへのアクセスがともにリモートログインである場合に限定されることになり，踏み台ホストから被害ホストへのアクセスがFTPのような場合は検出できない．また，検出に数十秒の時間が必要である．本論文では，このような踏み台攻撃を検出するため，踏み台ホストに対するリモートログイン操作の終了と同期して踏み台ホストから被害ホストに対してTCPコネクションの確立要求が送信されることに着目する．すなわち，踏み台ホスト宛のリモートログインパケットと，踏み台ホストから送信されるTCPのSYNパケットを監視することによって，リアルタイムに踏み台攻撃を検出するコネクションベース方式を提案する．提案方式をネットワークモニタ装置に実装して動作検証を行った結果，対象とする踏み台攻撃を確実に検出できることを確認した．

A Proposal of a Detection Technique on Stepping-stone Attacks Using Connection-based Method

Daisuke Takeo,[†] Masashi Ito,[†] Hidekazu Suzuki,[†]
Naonobu Okazaki[††] and Akira Watanabe[†]

Attackers usually use Stepping-stone attacks in order to hide their identity. To detect these kinds of attacks, timing-based method is conventionally used that detects the correlation of the traffic between receiving packets and sending packets at the Stepping-stone. However, this method can detect only remote-login chains and it takes tens of seconds. In this paper, we focus on the fact that a Stepping-stone always begins a TCP connection after receiving command sequence from the attacker. We propose a connection-based method that detects two specific TCP packets are within a certain period of time. The two packets are, a TCP SYN packet which is transmitted from a Stepping-stone to a target, and a remote-login packet that is transmitted from an attacker to a Stepping-stone. The proposed method can detect not only remote-login chains, but all kinds of Stepping-stone attacks immediately. We implemented our proposed connection-based method in a network monitor device and confirmed that the device effectively detect the attacks.

1. はじめに

企業ネットワークに対する不正アクセスはますます増加する傾向にある．外部からの不正アクセスに対しては，ファイアウォールやIDSなどのセキュリティ機器の導入や，つねに最新のセキュリティパッチを適用したり，不要なサービスを起動しないなど，ホストの要塞化によるセキュリティ対策がとられている．しかし，不正に入手したアカウント/パスワードを用いて被害ホストにアクセスするような攻撃を防ぐことは困難である．特にこのような攻撃を踏み台ホストを介して実行されると（以下，踏み台攻撃），管理者は攻撃者の身元を特定することすらできない．本論文では，リモートログインで遠隔地のホストにログインし，上記ホストを経由して，さらに別の遠隔地にあるホストへの攻撃を行う行為を踏み台攻撃と呼び，経由されたホストを踏み台ホストと定義する．ここでリモートログインとは，ログイン先のホストを手元にあるように操作ができるコマンドを指し，Telnet[1)]，SSH，Rloginなどがある．リモートログインを用いれば，遠隔ホストのファイル構造や内容を参照してインタラクティブ

† 名城大学大学院理工学研究科
Graduate School of Science and Technology, Meijo University
†† 宮崎大学工学部
Faculty of Engineering, University of Miyazaki

に次のアクションを決めることができる．リモートログインを，踏み台ホストを経由して実行すると，身元を隠蔽することが可能であり，被害ホスト側からはアクセス権を持った正常なユーザに操作されているようにしか見えない．リモートログインをいくつか経由すると（リモートログインの連鎖），真の攻撃者を特定することはきわめて困難となる．リモートログインによる攻撃は，試行錯誤による root 権限の奪取や，ログファイルの都合の悪い部分のみ改ざんするなどが可能であり，管理者側から見ると最も危険でかつ対処の難しい攻撃といえる．このような背景より，本論文では，リモートログインによる踏み台攻撃の存在を検出することを目的とする．攻撃タイミングを意図的に遅延する攻撃方法も存在するが，インタラクティブ性のある踏み台攻撃に比較すると危険度は低いため，今回は検討の対象からはずした．管理者は管理目的のためにサーバ類にはリモートログインサービスを提供している場合が多いため，安易にサービスを停止させることはできない．しかも単なるリモートログインやリモートログインの連鎖は正常な行為であり，既存のセキュリティ対策だけで踏み台攻撃を検出・防御することは難しいのが現状である．このようなことから管理者の意図に反して特定のホストが踏み台にされていることを検出することができれば有効である．

リモートログインによる踏み台攻撃の検出に注力した研究には以下のようなものがある．いずれも攻撃ホストから踏み台ホストを経由して被害ホストに至るすべてのアクセスがリモートログインである場合に限定されている．リモートログインを実行するために使用されるパケットをここではリモートログインパケットと呼ぶ．検出の手がかりにするものによって，コンテンツベース方式とタイミングベース方式の 2 種類に大別することができる．コンテンツベース方式には，リモートログインパケットのデータを直接比較する手法[2] や，リモートログインパケットに透かしを埋め込み，その透かしを比較する手法[3] があり，いずれも踏み台ホストをはさんだ 2 つのリモートログインパケットのデータ内容が一致していることを検出する．この方式は検出にパケットの内容やサイズを手がかりにしているため，暗号化されたリモートログインに対応できないという課題がある．一方，タイミングベース方式[4)~16)] はリモートログインのキー入力ストロークには特徴があることに着目しており，踏み台ホストをはさんだ 2 つのリモートログインストロークに時間的な相関関係があることを検出する．検出にはパケットの到着時間情報を手がかりにしており，パケットの中身やサイズに依存しないため，暗号化されたリモートログインにも対応可能である．現在はタイミングベース方式を採用した研究が主流であり，検出精度を高めるための様々な工夫が提案されている．しかし，これらの手法はいずれもリモートログインが連鎖状態にあることを検出することが基本で，被害ホストへの最終アクセスが FTP のようなリモートログイン以外の場合は検出の対象外である．また，タイミングベース方式は，相関関係を見る関係上，数十秒の検出時間を必要とするという課題がある．

本研究では，上記のような踏み台攻撃検出の一手法として，踏み台ホストに対するリモートログイン操作の終了と同期して，踏み台ホストから被害ホストに対して送信される TCP コネクション確立要求を検出するコネクションベース方式を提案する．この手法は，リモートログインパケットによる一連のコマンド情報を踏み台ホストが受信した直後に，踏み台ホストから被害者ホストに対して TCP コネクションの確立を要求するパケットが送信されることに着目したものである．リモートログインは，踏み台ホストに対して直接コマンドを投入しているのと同じであるため[1)]，遠隔ホストに対するアクセスコマンド情報を投入し終わるとその直後に対応した通信を開始することを利用する．提案方式を用いると，リモートログインが暗号化されていた場合や，被害者ホストへのアクセスがリモートログイン以外の様々なケースにおいても確実に踏み台攻撃の検出が可能になる．また，提案方式は特定の通信パケットをリアルタイムで監視するため，踏み台攻撃が発生した直後にそのことを検出することができる．

ネットワーク型監視ホストとして提案方式を実装し，動作確認を行った結果，踏み台攻撃を確実に検出できることを確認した．ネットワーク管理者はサーバが踏み台にされていることを早期に知ることができるため，提案方式は不正アクセスを防止する有効な手段となりうる．多くのサーバでは管理者が管理目的のために Telnet などのサービスを故意に稼動させている場合があり，踏み台ホストにされる可能性が高い．また，当該サーバが攻撃対象となる装置にすでにコネクションを張っていたとしても，新たなコネクション要求があれば改めてコネクションを張りにゆく．したがって，本提案方式は踏み台ホストが恒常的に外部にコネクションを張る役割を持つ場合においても有効と考えられる．

以降，2 章で踏み台攻撃の定義と関連研究の紹介をし，3 章で提案方式の概要を説明する．4 章では実装について述べ，5 章で評価を行う．最後に 6 章でまと

める．

2. 従来の踏み台攻撃検出方式

2.1 踏み台攻撃の定義

本論文において対象とする踏み台攻撃モデルを**図 1** に示す．踏み台攻撃モデルの構成要素として，Attacker，Stepping-stone，Target を定義する．Attacker（攻撃ホスト）は攻撃者が直接操作するホストである．Stepping-stone（踏み台ホスト）は Attacker がリモートログインプロトコルを用いてアクセスするホストである．Target（被害ホスト）は Attacker が Stepping-stone を介してアクセスするホストである．Attacker は何らかの方法を用いて，あらかじめ Stepping-stone および Target のアカウントとパスワードを入手しているものとする．複数の Stepping-stone を経由した攻撃もあるが，検出の原理は同じであるため，本論文では Stepping-stone が 1 台の場合について検討する．Stepping-stone から Target までのアクセス方法は必ずしもリモートログインの必要はなく，FTP のようなプロトコルを用いることも可能である．このように，Stepping-stone を経由して Target にアクセスする攻撃を，ここでは踏み台攻撃と呼ぶ．Attacker から Stepping-stone へのアクセスのフローを Flow X，Stepping-stone から Target へのアクセスのフローを Flow Y と呼ぶ．ここでは特に，Flow X がリモートログインである場合を対象とする．リモートログインのプロトコルとしては，Telnet，SSH，Rlogin がある．

2.2 従来研究とその課題

従来の踏み台攻撃検出方式は，Flow X および Flow Y がいずれもリモートログインである場合にのみ適用が可能である．初期の方式として，Flow X と Flow Y を流れるリモートログインパケットの内容を手がかりにするコンテンツベース方式が提案されている．

文献 2) では，踏み台攻撃が行われるとき，パケットのデータ内容が同一であるリモートログインパケットが Flow X と Flow Y を流れることに着目しており，Flow X と Flow Y の一連のパケットの内容を比較して，一致していることを検出する．

文献 3) では，Target に対してリモートログインによる不正侵入が行われた場合，Target から Attacker へと戻るリモートログインパケットに透かしを埋め込んでおき，Flow X と Flow Y で同じ透かしが埋め込まれていることを検出する．Stepping-stone が踏み台にされた時点で踏み台攻撃を検出するのではなく，Target に対して不正侵入が行われたことを検出してからでなければ検出できない．

コンテンツベース方式はリモートログインパケットの内容を見る必要があるため，パケットが暗号化されると適用できない．このことから近年ではタイミングベース方式による検出が主流となっている．タイミングベース方式では，ユーザがリモートログインを利用するときのキー入力ストロークには特徴があることに着目しており，Flow X と Flow Y のリモートログインストロークの間に時間的な相関関係があることを検出する．

たとえば**図 2** のように，ある期間内に Stepping-stone 前後で Flow X と Flow Y が発生したとき，各フローのパケット発生タイミングを手がかりにし，2 つのフローに時間的な相関関係があるかどうかを発見する．この方式はパケットの内容を識別する必要がないため，SSH など暗号化されたリモートログインにも対応可能である．

文献 4) の ON/OFF ベースアプローチでは，トラフィックが発生していない期間，すなわちフローの各パケットの送信間隔時間を手がかりにする．2 つのフローの各パケット送信間隔時間が同じであれば，踏み台攻撃であると判断する．しかし，あらゆるケースで検出率をあげるのは難しく，ネットワークトラヒックなどによって検出に要する適切なパラメータ値が異なるという課題がある．

ON/OFF ベースアプローチ以外でも，検出精度やリアルタイム性の向上を図った手法，踏み台の連鎖がループ状になった踏み台攻撃の検出手法，踏み台の連鎖の長さを推測する手法など，踏み台攻撃検出に注力した研究が行われている．

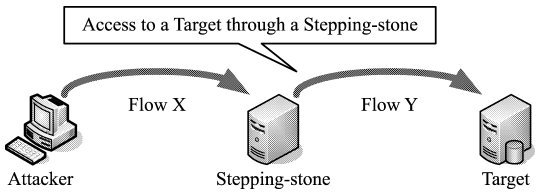

図 1 踏み台攻撃モデル
Fig. 1 A stepping-stone attack model.

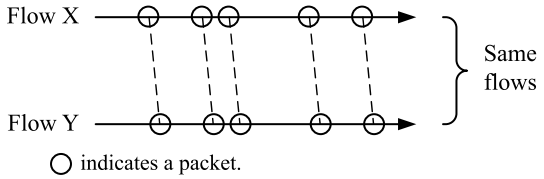

図 2 相関関係の発見
Fig. 2 Discovery of the correlation between two flows.

表 1 タイミングベース方式の研究の分類
Table 1 A classified table of timing-based methods.

文献番号	内容
4)，6)	ON/OFF の相関関係から推測する提案
5)	踏み台ホストでの平均遅延時間から推測する提案
7)	ウォーターマークを併用して検出精度を向上させる提案
8)，9)	踏み台攻撃の連鎖をトレースバックする方式の検討
10)	踏み台攻撃の検出に必要となるパケット数を定量化
11)，16)	パケット遅延やパケット挿入があっても検出精度を維持する
12)	リモートログイン経路がループしていた場合の探索可能性
13)〜15)	リモートログインのラウンドトリップ時間より連鎖の長さを推測

これらの方式は Target へのアクセスもリモートログインである場合に限定されており，Target に対してそれ以外のプロトコルを利用するような踏み台攻撃を検出することはできない．また，Flow X と Flow Y が所定の時間を経過しないと相関関係があると判断することができない．攻撃者はログインストロークの間にディレイや余計なパケットを挿入するなどにより，相関関係の検出を困難にすることができることが指摘されている．**表 1** にこれらの研究を分類したものを示す[4)〜16)]．

3. コネクションベース方式の提案

前述のように従来の踏み台攻撃検出手法は，Flow X と Flow Y がともにリモートログインを用いた場合にのみ適用可能である．しかし，実際には Target に対するアクセスがリモートログインだけとは限らない．たとえば FTP によりファイルをダウンロード/アップロードすることも考えられる．ここで，Attacker が Stepping-stone へとリモートログインした後，そこから Target に対してあらゆる TCP 通信でアクセスする可能性があることを想定すると，踏み台攻撃発生時には必ず Stepping-stone から TCP コネクションの確立が必要である．また，その直前には TCP コネクション確立要求送信のトリガとなるコマンドを乗せたリモートログインパケットを Stepping-stone が受信しているはずである．本論文では，Stepping-stone へのリモートログインパケットが送信されたのち，一定時間内に Stepping-stone から TCP コネクションが確立されることを検出することにより踏み台攻撃を検出するコネクションベース方式を提案する．

コネクションベース方式では，Stepping-stone が存在するネットワーク上に通信を監視するホスト（以下，Detector と呼ぶ）を設置する（**図 3**）．Detector を設置したネットワークに存在するホストはどれが Stepping-stone であってもかまわない．

図 4 にコネクションベース方式の原理を示す．ま

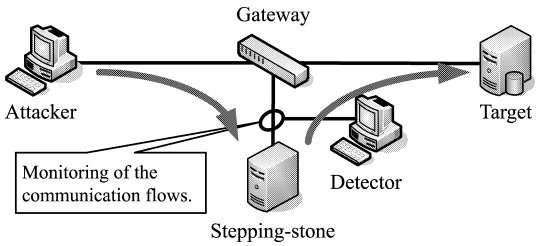

図 3 Detector の設置箇所
Fig. 3 Installation place of Detector.

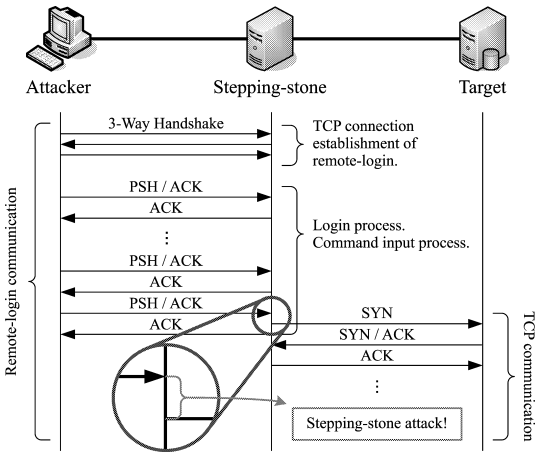

図 4 コネクションベース方式の原理
Fig. 4 The principle of Connection Base Method.

ず Attacker が Stepping-stone へリモートログインするために TCP コネクションの確立を行う．次に，Attacker は Stepping-stone に対して Target へのアクセスを行うためのコマンドを投入する．コマンドの最後の文字が入力されると，Stepping-stone はコマンドを解読して，Target に対して TCP コネクションの確立を行う．Detector はその間リモートログインパケットの監視を行いつつ，他のホストへ新たな TCP コネクションが確立されようとするのを監視する．リモートログインパケットの受信とコネクション確立要求の送信との間の時間が一定時間内であれば，

Detector はこの状況を踏み台攻撃と判断する．

以上の原理に従い，コネクションベース方式では Detector 上でキャプチャしたパケットに対して以下の処理を行う．

(1) **リモートログインパケットの監視**

TCP パケットのうち，Telnet，SSH，Rlogin などのリモートログインの通信パケットでかつ PSH ビット[17]のセットされたパケットを検出する．検出するたびにパケットの送信元/宛先 IP アドレスと送信元/宛先ポート番号，検出した時刻をリモートログイン受信記録として保存していく．PSH ビットを見る理由は，リモートログインの通信パケットはローカルホストで入力された 1 文字分（あるいは 1 行分）のデータであり，受信したらすぐにアプリケーションに渡すため，PSH ビットが必ずセットされているからである．TCP [17] と RFC1122 [18] の規定によると，送信側 TCP はデータをいつまでもバッファに溜め込んではならず，また，最後にバッファに溜めたセグメントには PSH ビットをセットしなければならないとされている．また，受信側 TCP は PSH ビットがセットされていたパケットを受信した場合は，強制的にアプリケーションにデータを渡すものとされている[1]．したがって，送信側からリモートログインコマンドの最後の文字（一般にはリターンコード）を含むパケットが送信される場合には必ず PSH ビットがセットされており，受信側ではこれを受信するとただちにアプリケーションでのコマンドの解読が開始される．

(2) **TCP コネクション確立要求パケットの監視**

TCP パケットのうち，あらゆる TCP サービスの SYN パケットを検出する．SYN パケット検出時にリモートログイン受信記録を参照し，リモートログインパケットの宛先 IP アドレスと SYN パケットの送信元 IP アドレスが一致するものを検索する．一致するものがあれば，リモートログインパケットの検出時刻と SYN パケット検出時刻とを比較し，一定時間内に SYN パケットの送信が行われていれば，Stepping-stone が踏み台になっているものと判断する．SYN フラグを見る理由は，Attacker からのコマンドを受けて Stepping-stone が Target に対して TCP サービスを起動する場合，必ず初めにコネクション確立要求を送信するので，これを検出すればよいからである．以降，リモートログインの PSH ビットの立ったパケットを検出してから TCP の SYN パケットを検出するまでの時間を踏み台検出時間あるいは単に検出時間と呼ぶ．また，踏み台検出時間になっていると判断するまでの最大待ち時間のことを監視時間と呼ぶ．検出時間が監視時間を超えている場合は，踏み台攻撃ではないと判断する．コネクションベース方式ではリモートログインパケットのデータ内容を参照する必要がないので，リモートアクセスが暗号化されていてもかまわない．また，Target 宛送信パケットの SYN フラグだけを検出すればよいため，Stepping-stone から Target へのあらゆる TCP 通信を検出対象とすることができ，かつ踏み台攻撃が発生したことを即座に検出できる．

4. 実　　装

コネクションベース方式を実現するためには，ネットワークを流れる全通信パケットの監視を行う必要がある．パケットの監視は libpcap でキャプチャすることとし，FreeBSD 上で動作するアプリケーションとして本方式を実現した．

コネクションベース方式による踏み台攻撃検出プログラムの試作モジュール構成図を**図 5** に示す．

ネットワークインタフェースが受信した全パケットのうち，TCP パケットのみを検出プログラムに渡すように libpcap にフィルタを掛ける．検出プログラムは，パケットがリモートログイン通信であれば，受信記録追加モジュールにパケットを渡す．TCP コネクション確立要求であれば，受信記録検索モジュールにパケットを渡す．その他の場合は無視し，次のパケットの受信を待つ．

受信記録追加モジュールは，リモートログインパケットの送信元 IP アドレス，宛先 IP アドレス，送信元ポート番号，宛先ポート番号，受信時刻からなるレコードを，リングバッファにリモートログイン受信記録として保存する．

受信記録検索モジュールは，リモートログイン受信記録から，TCP コネクション確立要求パケットの送信元 IP アドレスと一致する宛先 IP アドレスを持つレコードを検索する．IP アドレスが一致するものが

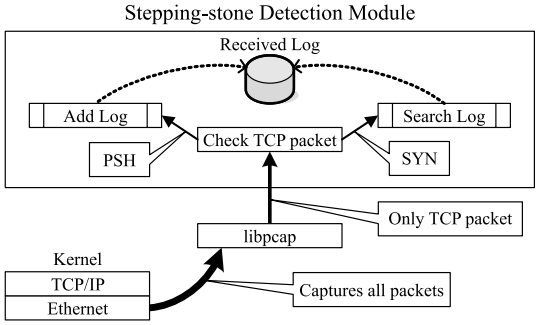

図 5 試作モジュール構成図
Fig. 5 Trial module construction.

検出され，さらにリモートログインパケットの受信時刻が TCP コネクション確立要求パケットの送信時刻からさかのぼって監視時間内にあれば，踏み台攻撃が発生したことを示すメッセージを出力する．

本方式はパケットの IP ヘッダと TCP ヘッダの一部，およびリモートログイン受信記録を参照するだけでよい．試作プログラムのステップ数は 270 ステップ程度であり，処理内容はきわめて単純である．

5. 評　　価

5.1 評価の概要

上記で述べたように，監視時間は様々なケースにおける踏み台検出時間のばらつきを考慮して設定する必要がある．そこで，ここでは踏み台検出時間のばらつきを明らかにするために，2 つの評価実験を行った．まず，Stepping-stone におけるアプリケーションの動作の状態や通信トラフィックの状態による影響を調べるための基礎的な実験として，Stepping-stone に背景負荷を与えた場合の踏み台検出時間の測定を行った（評価実験 1）．次に，より実際の環境における踏み台検出時間のばらつきを調べるために，実際に運用されているサーバを Stepping-stone とした場合の踏み台検出時間の測定を行った（評価実験 2）．

踏み台攻撃は，Attacker が Stepping-stone にまず Telnet などのリモートログインで侵入する．この時点で Stepping-stone 側でコマンドに対応する daemon が起動される．Attacker は Stepping-stone をあたかも手元にあるように操作して，次のホストに侵入することができる．実際の攻撃では Stepping-stone が何段か経由することもあるが，本実験では，Stepping-stone の動作を解析するのが目的のため，Stepping-stone は 1 台とした．Attacker がコマンドの最後の文字（一般にはリターンコード）を投入すると，PSH ビットの立ったパケットが Stepping-stone に送信される．Stepping-stone が当該コマンドを受信するとただちにコマンドの解読を開始し，Target へ通信パケットを送信する．このとき，Target へ送信される最初のパケットは TCP のコネクション要求であり，TCP SYN ビットが必ずセットされている．本実験では，PSH ビットの立ったパケットが Stepping-stone に到着したときの時間と，TCP の SYN パケットが送信されたときの時間の間隔を踏み台検出時間として測定した．Flow Y としては，あらゆる TCP 系のコマンドが可能であるが，ここではリモートログインの代表として Telnet，それ以外のコマンドの代表として FTP を測定した．ここで示した評価モデルは，リモートログインによるアクセスを隠蔽する行為としてよくある攻撃者の挙動である[4),10)]．

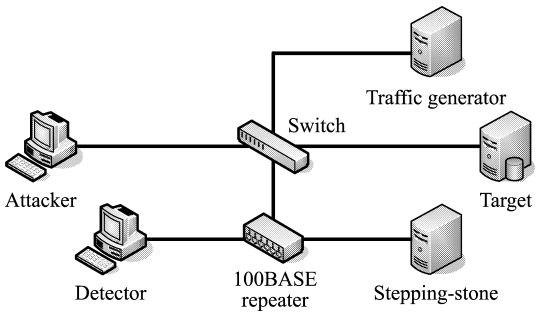

図 6 評価実験 1 の環境
Fig. 6 Environment for experiment 1.

表 2 評価実験 1 で用いた端末の仕様
Table 2 Specification of the terminals for experiment 1.

	Detector	Stepping-stone
CPU	Celeron 1.7 GHz	Pentium4 2.4 GHz
RAM	256 MB	256 MB
OS	FreeBSD	FreeBSD
	5.3-RELEASE	5.3-RELEASE

5.2 評価実験 1

評価実験 1 では，Stepping-stone に背景負荷を与えた場合の踏み台検出時間を測定した．評価実験 1 の評価環境を**図 6** に示す．踏み台攻撃モデルを構成する Attacker，Stepping-stone，Target に加えて，コネクションベース方式による踏み台攻撃検出プログラムを実装した Detector とトラヒック発生装置を用意した．Stepping-stone と Detector は 100BASE のリピータで接続し，Detector が Stepping-stone に流れる Flow X と Flow Y の両方の通信を監視できるようにした．それ以外の機器はスイッチを用いて接続した．Stepping-stone では Telnet，SSH，Rlogin，FTP サービスを，Target では Telnet，FTP サービスを起動させた．

Detector，Stepping-stone の装置仕様は**表 2** のとおりである．Attacker から Stepping-stone へのリモートログインには Telnet，SSH，Rlogin を，Stepping-stone から Target への TCP 通信には Telnet，FTP を用い，いずれの場合においても踏み台動作を確実に検出できることを確認した．Flow Y については，リモートログイン以外のプロトコルとして FTP のみを選択したが，HTTP や HTTPS などは同じ TCP 上のアプリケーションであり，TCP SYN の動作は変わらないため，同様な結果が得られるものと判断した．なお，本実験では DNS を使用せず，直接 IP アドレス指定で行った．

表 3 踏み台検出時間測定結果（背景負荷なし）
Table 3 Results of stepping-stone detection time (Without background load).

		Flow Y			
		Telnet		FTP	
		(1)	(2)	(1)	(2)
Flow X	Telnet	7.69	2.55	3.81	0.35
	SSH	7.73	2.62	3.92	0.39
	Rlogin	7.61	2.53	3.78	0.33

* (1) 起動時接続先指定 (2) 起動後接続先指定
値は 10 回試行の平均（単位：ms）

5.3 評価実験 1 の測定結果

踏み台検出時間を Stepping-stone に背景負荷を与えない状態で測定した結果を**表 3** に示す．同表において起動時接続先指定とは，接続先を指定するパラメータとして Target を指定して Telnet クライアントまたは FTP クライアントを起動して接続した場合であり，起動後接続先指定とは，Telnet クライアントまたは FTP クライアントを起動した後に内部コマンドを用いて接続先を Target に指定して接続した場合である．Target への接続方法を 2 通り試行した理由は，まったく同じ TCP のクライアントプログラムを用いても異なる挙動をすることを調査するためであり，起動時接続先指定は起動後接続先指定に比べてプログラムを起動する処理の分だけオーバヘッドが多い．

Flow X に用いたリモートログインの種類によって若干異なるものの，Flow Y が Telnet 起動時接続先指定では 7.6〜7.8 ms 程度，Telnet 起動後接続先指定では 2.5〜2.7 ms 程度の間に SYN パケットが送信されていることが分かる．また Flow Y が FTP 起動時接続先指定では 3.7〜4.0 ms 程度，FTP 起動後接続先指定では 0.3〜0.4 ms 程度となり，Flow Y が Telnet である場合より短いという結果が得られた．

さらに，踏み台検出時間が変動する要因を調査するために，Stepping-stone に対してトラヒック発生装置から FTP 接続または HTTP 接続を実行し，ファイル転送により Stepping-stone に背景負荷を連続的に与えた状態で踏み台検出時間を測定した．負荷となる FTP および HTTP のファイル転送量を増加させていき，Flow X が Telnet，Flow Y が Telnet と FTP である場合の踏み台検出時間の測定を行った．

図 7 に FTP を背景負荷としてファイル転送量を変えた場合の踏み台検出時間を，**図 8** に HTTP を背景負荷としてファイル転送量を変えた場合の踏み台検出時間を示す．図の横軸はファイル転送のセッション数を，縦軸は踏み台検出時間を表している．

測定結果より，ファイル転送のセッション数が増加

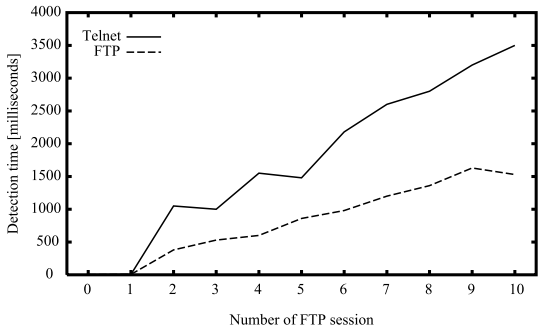

図 7 検出時間と FTP 背景負荷の関係
Fig. 7 Detection time vs. FTP traffic.

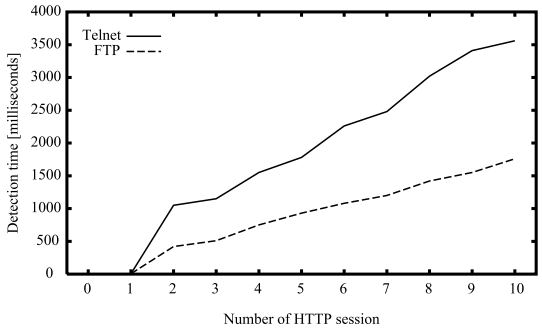

図 8 検出時間と HTTP 背景負荷の関係
Fig. 8 Detection time vs. HTTP traffic.

するにつれて踏み台検出時間がほぼ比例して増加していることが分かる．Flow Y が Telnet と FTP の場合では，Telnet の方が踏み台検出時間に倍程度の時間を要している．背景負荷が FTP と HTTP の場合では，どちらのときであってもほぼ同様の検出時間の増加が見られた．このときの Stepping-stone の CPU 負荷を見ると，FTP および HTTP の場合ともセッション数が 2 程度で飽和し，FTP の場合 85%，HTTP の場合 25%という結果が得られた．この結果から，検出時間の増加は Stepping-stone の CPU 負荷が原因ではなく，LAN カードにかかる通信負荷が影響しているものと想定される．すなわち，同一のセッション数では FTP と HTTP の通信負荷同程度であるため，検出時間も同程度の影響を受けたものと思われる．

5.4 評価実験 2

評価実験 2 では，実際に学内で運用されているサーバを Stepping-stone にみたて，どのような踏み台検出時間が得られるかを測定した．評価実験 2 の評価環境を**図 9** に示す．今回選定した Stepping-stone は，Telnet サービスが学内のメンバに公開されている外部・内部接続用中継サーバで，グローバル IP アドレスが割り当てられている．この中継サーバに研究室内に

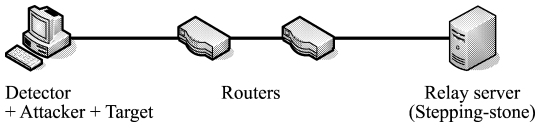

図 9 評価実験 2 の環境
Fig. 9 Environment for experiment 2.

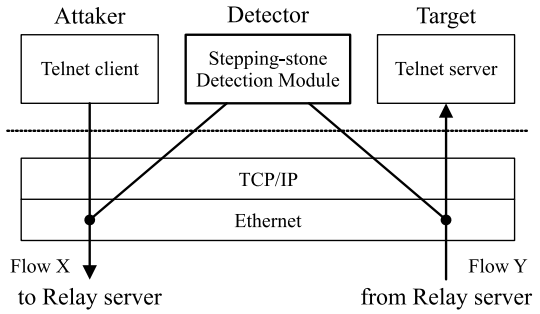

図 10 評価装置の構成
Fig. 10 Construction of the evaluation device.

設置した Detector から学内ネットワークを通じてリモートログインによるアクセスを行い，さらに当該中継サーバから Detector に対してリモートログインするという手順で擬似的に踏み台攻撃を行った．本来であれば Detector は Stepping-stone と同じネットワークに設置するべきであるが，Detector を設置するにはリピータを接続するなど，サーバ室内の機器構成の接続を変更する必要がある．このため，同様の環境を模擬するために，図 9 のように評価装置が Detector だけでなく，Attacker と Target の役割を兼用することにより，Stepping-stone となる中継サーバに流れる Flow X と Flow Y の両方の通信を監視できるようにした．評価装置は**図 10** のような構成となっている．評価装置では Target となる Telnet サーバと，本提案方式を実装したモジュール（図 5 で示したもの）を Detector として起動しておく．この状態で Telnet クライアントにより中継サーバにリモートログインし，さらに中継サーバを踏み台にして評価装置（自分自身）にリモートログインする．検出モジュールは，評価装置自身が送受信するリモートログインパケットを監視する．この動作を繰り返すことにより，検出モジュールが Telnet アプリケーションとは独立して踏み台検出時間のデータを収集した．

中継サーバ（Stepping-stone）は学内ですでに稼働運用されているサーバであり，Linux で実現されている．中継サーバと評価装置間に介在するルータ数は 2 台であった．中継サーバと評価装置の仕様は**表 4** のとおりである．Flow X と Flow Y にはともに Telnet を使用した．実験を行った期間は日曜日から土曜日まで

表 4 評価実験 2 で用いた端末の仕様
Table 4 Specification of the terminals for experiment 2.

	Detector	Stepping-stone
CPU	Celeron 1.7 GHz	Xeon 3.06 GHz
RAM	256 MB	2,048 MB
OS	FreeBSD	Red Hat Enterprise
	5.3-RELEASE	Linux ES release 3

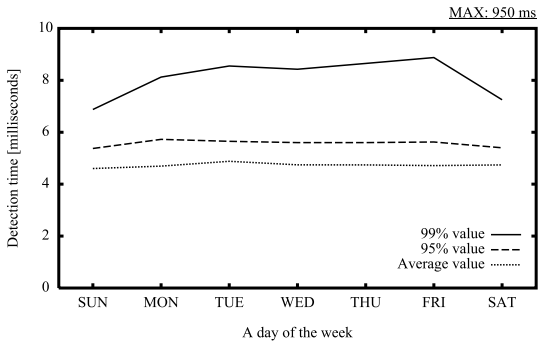

図 11 1 週間の検出時間の推移
Fig. 11 Changes in detection time within a week.

での 1 週間で，1 時間あたり 480 回の擬似踏み台攻撃を連続的に試行し，踏み台検出時間の測定を行った．

5.5 評価実験 2 の測定結果

中継サーバに対する 1 週間の検出時間の推移を**図 11** に示す．図の横軸は曜日を，縦軸は検出時間を表しており，1 日ごとに得られた踏み台検出時間の平均値，95%値，および 99%値を示す．ここで n%値とは，検出回数の n%がその時間内に入る値である．測定結果から，平均値は 4.7 ms 程度で推移し，99%が 10 ms 以下であることが分かる．しかし，最大値として 950 ms という値が観測されており，頻度は少ないが大きな検出時間になる場合がありうることが分かる．平均および 95%値は曜日による差はさほどないが，99%値では土曜日，日曜日の反応時間が平日より少なく，休日はサーバが混雑していないことがうかがえる．

さらに実験期間のうち，平日の水曜日について，踏み台検出時間の 1 日の推移を示したものを**図 12** に示す．図の横軸は時刻を，縦軸は検出時間を表しており，1 時間ごとに得られた踏み台検出時間の平均値，95%値，および 99%値を示している．図 11 と同様に，平均値，95%値では時間帯による変動はさして大きくないが，99%値は昼間に大きくなっており，特に 18〜19 時ごろに顕著な値を示している．99%値が大きい時間帯はサーバへのアクセスが多いものと推測できる．当日の最大値は 210 ms（19 時）であり，数は少ないがやはり極端に大きな検出時間が観測されていることが分かる．

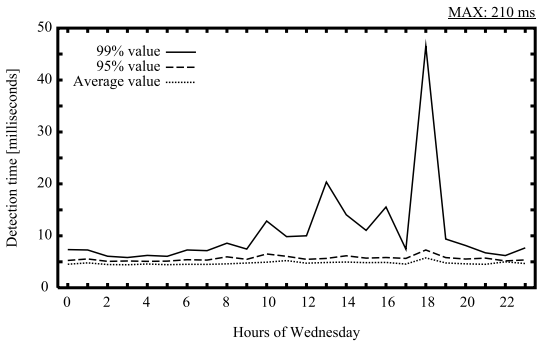

図 12 1 日の検出時間の推移
Fig. 12 Changes in detection time within a day.

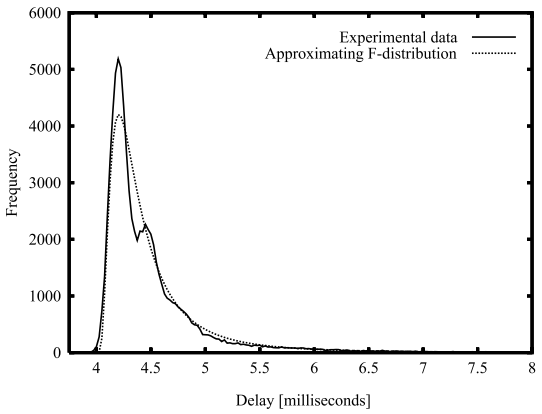

図 13 踏み台検出時間の分布と F 分布による近似
Fig. 13 Distribution of experimental detecting times and approximating F-distribution.

次に全期間中における検出時間の分布を**図 13** に示す．検出時間を 25 μs に区切り，各期間に属するデータの個数を縦軸に示した．4 ms 以下のデータは存在せず，踏み台処理に必ず所定の時間が必要なことが分かる．ここで所定の時間とは，Stepping-stone において，リモートログインで指示された命令を解読し，その指示に従って TCP SYN パケットを生成するために必要となるプログラムの処理（パケット受信処理 + コマンド解析処理 + SYN パケット送信処理）である．10 ms 以上の時間は頻度としては少ないが，わずかなデータが横軸上に長く存在し，最大値が 950 ms となる．

踏み台検出時間の分布は複数の事象が複雑にからみあっている．考えられる要因としては，Stepping-stone の CPU 負荷，データ転送負荷のほか，ネットワークのトラフィック負荷，経由するルータの処理負荷などがある．図 7，図 8 から，Stepping-stone に与えられる背景負荷により踏み台検出時間が大きく変動することが分かっているので，これが 950 ms や 210 ms という，まれであるが大きな値になる最大の要因になっていることが推測できる．

表 5 踏み台検出時間の n%値
Table 5 N% value of detection times.

踏み台検出時間	平均	90%値	99%値	99.9%値
実測値	4.8	5.0	8.1	60.1
近似 F 分布	4.8	5.0	7.1	12.4

（単位：ms）

実験で得られたデータを様々な確率分布にあてはめてみたところ，自由度（13, 5）の F 分布に最も近いという結果が得られた．ここで F 分布の自由度の値が大きいほど多くの事象がからんでいることを示している．図 13 に近似した F 分布を実測値とともに示す．実線が実測値，点線が近似した F 分布である．図 13 から n 値を求めたところ，**表 5** のようになった．n が 99 程度であれば実測値とよく一致するが，100 に近づくと大きな開きが出てくる．踏み台検出時間のばらつきは，からみあう事象が多岐にわたるため，確率分布を予測することは難しい．今回は結果的に F 分布によって最も近似できたということである．このようなことから，監視時間は，実測値に基づいてシステムごとに決定するのが適切と思われる．評価環境 2 の評価環境での検出率をほぼ 100%にするためには，今回の実測値より監視時間を 1 秒程度にすればよいといえる．

評価実験 2 で測定した踏み台検出時間は，正確には Stepping-stone の純粋な踏み台検出時間に，ルータ 2 段の中継時間を加えた値となる．2 台のルータが加わったことによる時間の増加は，通常は十分小さいものと推定できるが（200 μs 以下），ネットワーク負荷が大きくなったときにどの程度の影響を与えるかを推測するのは困難である．評価実験 2 では，この値はつねに十分小さいものと仮定して考察を行っている．

また，本実験では直接 IP アドレス指定で行っているため，ドメイン名を用いる場合は DNS lookup の時間が余分にかかる．この時間は，途中のネームサーバにキャッシュがない場合は約 350 ms，キャッシュがある場合は約 15 ms という結果が示されている（いずれも平均値）[19]．また，DNS query/response が伝送速度の遅いネットワークを通る場合は，伝送速度の影響をさらに考慮する必要がある．DNS パケット長（DNS query+DNS response）を 300 バイトとすると，1 Mbps ADSL であれば 1 つの伝送線路上で 1 問合せあたり 2.4 ms，8 Mbps ADSL であれば 300 μs，100 Mbps FTTH であれば 24 μs の時間を要する．このように実環境においては，様々な要因が踏み台検出

時間の変動にかかわってくる．

5.6 コネクションベース方式の課題

コネクションベース方式の課題としては，監視対象パケットの発生時刻の時間差を手がかりとしているため，大量の監視対象パケットが発生すると Attacker の特定が困難になる可能性がある．そこで，踏み台攻撃検出機能で Attacker を特定せず，疑わしい Attacker のリストを管理者に報告するなど，最終的な判断は人間に任せる必要があると考えられる．

評価実験 1 の結果から，Stepping-stone に対する背景負荷が大きい状態において，踏み台攻撃と判断するまでの時間の閾値である監視時間が短いと，踏み台攻撃が検出できない可能性があると考えられる（false negative の増加）．逆に監視時間を長くすれば必ず検出できるが，一定時間内にネットワーク上を流れるリモートログインパケットが多くなり，Attacker の候補が増加することによって Attacker を特定しにくくなる可能性がある（false positive の増加）．

ここで，false positive としては 2 種類のケースを考える必要がある．1 つ目は，正規のユーザがリモートアクセスの連鎖を行う場合を区別する必要がある．UNIX に慣れたユーザは，リモートアクセスで複数のホストを経由してメールサーバなどにアクセスすることを一般に行う．そこで，このような通常の処理におけるリモートアクセスの連鎖が誤検出の対象とならないように，ありうる連鎖のリストをあらかじめ登録しておき，このようなユーザが誤検出の対象とならないようにする必要がある．2 つ目は，リモートログインが頻繁に実行される環境において，踏み台攻撃の連鎖とそうでない連鎖が時間的に近接して誤検出を引き起こす場合が考えられる．監視時間が必要以上に長いと，false positive の原因となるだけでなく，メモリや CPU 資源を無駄に浪費する．したがって，監視時間はリモートログインの PSH ビットがセットされたパケットが Stepping-stone により受信されてから，Target への攻撃を行う SYN パケットが送信されるまでの時間をあらかじめ見込んだうえで，必要最小限に設定する必要がある．この時間は，Stepping-stone の性能や他のアプリケーションの動作による CPU やメモリの利用状態，通信トラフィックの状態によって異なると考えられる．具体的には，監視対象となるホストの存在するネットワーク上で評価実験 2 に示すようなテストデータを取得して決定することが望ましい．Stepping-stone に対する背景負荷の量（通信量や CPU 負荷など）を同時に監視しておき，踏み台攻撃と判断するまでの監視時間を動的に設定するという方法も考えられる．

5.7 検出対象外となる踏み台攻撃

本方式では以下のようなケースは踏み台攻撃の検出対象とはしない．たとえば，Target 上で UDP によるサービスを起動している場合には，Target に対して，UDP による攻撃が行われる場合がありうる．提案方式は，Target に対して TCP コネクションを張るタイプの攻撃についてのみ検討対象とするため，このような攻撃は検出しない．したがって，本方式を適用する際には，人事情報や経理情報を格納するような重要なサーバ上ではこのような UDP サービスは起動しないなどの処置が必要である．また，事前に Stepping-stone に bot のようなリモートコントロールプログラムを仕込んでおき，たとえば Target に対して時間差をつけて攻撃をしかける方法も可能である．さらに，UNIX では指定時間だけコマンドの実行を停止できる Sleep というコマンドがあり，Attacker が Sleep を用いて攻撃に時間差をつける方法もありうる．提案方式では一定の監視時間を設けてこの時間を超えたものは検出しないようにしたため，これらの攻撃の検出は困難である．ただし，このような攻撃を行うためには攻撃側にも高度な技術と準備が必要で，事前にリモートログインなどを用いて Target を調査しておく必要があるため，この段階において提案方式が適用できる可能性がある．また，たとえばリモートコントロールプログラムを Stepping-stone に仕込む攻撃であれば，そのようなプログラムを仕込む機能を有したコンピュータウィルスやトロイの木馬プログラムを検出・駆除することで，事前に攻撃を防ぐなどの処置を行うことができる．

5.8 応 用 事 例

本提案方式を，IDS（Intrusion Detection System）の一機能として実装した例を**図 14** に示す．同図は，本提案方式を IDS に内蔵した場合のシステム構成である．企業の外部公開サーバがリモートログイン連鎖の経路となり，他システムのホストを攻撃している様子を示している．企業ネットワークでは，一般にファイアウォールのバリアセグメント上に IDS を設置し，外部公開サーバに対する攻撃を検知する．外部公開サーバがリモートログインによる踏み台攻撃の対象となった場合，現状の IDS 機能ではこれを検知する適切な手段がない．外部公開サーバは知らないうちに他ホストへの Stepping-stone となり，他サイトへの攻撃の加害者と見なされる恐れがある．IDS に本提案方式を内蔵することにより，企業の外部公開サーバが踏み台攻撃の対象になることを防止することが可能になる．

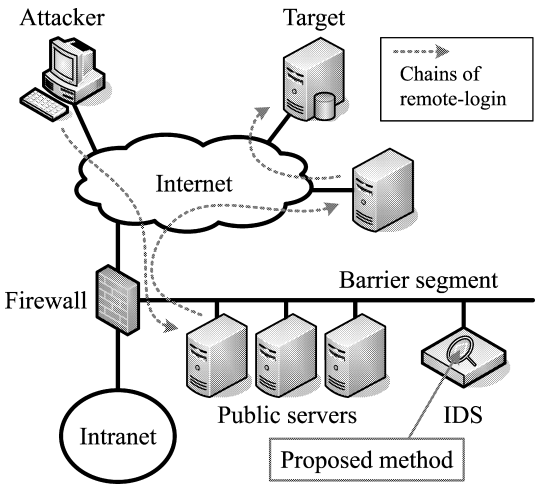

図 14 提案方式の応用例
Fig. 14 Application of the proposed method.

6. ま と め

本論文では，踏み台攻撃発生時に Stepping-stone から Target に対して TCP コネクション確立要求が必ず送信されることに着目し，Attacker から Stepping-stone に対するリモートログインパケットが発生してから一定時間内に TCP コネクション確立要求パケットが発生することを検出するコネクションベース方式を提案した．提案方式では TCP ヘッダのコントロールフラグを参照する方式であるため，Stepping-stone に対するリモートログインに用いられるプロトコルはどのようなものであってもよい．また Target に対するアクセスはどのような TCP 通信であってもかまわない．検出方法が TCP コントロールフラグを参照するだけの簡単なアルゴリズムであることから，踏み台攻撃をリアルタイムに検出することができる．

提案方式をネットワーク型機器として実装して動作確認を行い，踏み台攻撃を検出できることを示した．評価実験を行い，踏み台検出時間のデータを示した．そして，Stepping-stone に対する通信量や TCP 通信を行うプログラムの種類などによって踏み台検出時間が変化することを明らかにし，適切な監視時間の設定が必要であることを示した．今後は false positive の低減や，踏み台攻撃と正常な間接ログインをより精度高く識別する方法，攻撃者を適切に特定する方法などについて検討を進める予定である．

参 考 文 献

1) Postel, J. and Reynolds, J.: TELNET PROTOCOL SPECIFICATION, RFC 854, IETF (1983).
2) Staniford-Chen, S. and Heberlein, L.T.: Holding Intruders Accountable on the Internet, *Proc. 1995 IEEE Symposium on Security and Privacy* (*SP'05*), pp.39–49 (1995).
3) Wang, X., Reeves, D.S., Wu, S.F. and Yuill, J.: Sleepy Watermark Tracing: An Active Network-Based Intrusion Response Framework, *Proc. 16th International Conference on Information Security* (*IFIP/Sec'01*), pp.369–384 (2001).
4) Zhang, Y. and Paxson, V.: Detecting Stepping Stones, *Proc.9th USENIX Security Symposium*, pp.171–184 (2000).
5) Yoda, K. and Etoh, H.: Finding a Connection Chain for Tracing Intruders, *Proc.6th European Symposium on Research in Computer Security* (*ESORICS2000*), LNCS, Vol.1895, pp.191–205 (2000).
6) Wang, X., Reeves, D.S. and Wu, S.F.: Inter-Packet Delay Based Correlation for Tracing Encrypted Connections through Stepping Stones, *Proc. 7th European Symposium on Research in Computer Security* (*ESORICS2002*), LNCS, Vol.2502, pp.244–263 (2002).
7) Wang, X. and Reeves, D.S.: Robust Correlation of Encrypted Attack Traffic Through Stepping Stones by Manipulation of Interpacket Delays, *Proc. 10th ACM Conference on Computer and Communications Security* (*CCS2003*), pp.20–29 (2003).
8) Strayer, W.T., Jones, C.E. and Castineyra, I.: Dynamic Virtual Private Networks, Technical Report BBN Report 8384, BBN Technologies (2003).
9) Donoho, D., Flesia, A., Shankar, U., Paxson, V., Coit, J. and Staniford, S.: Multiscale Stepping-Stone Detection: Detecting Pairs of Jittered Interactive Streams by Exploiting Maximum Tolerable Delay, *Proc. 5th International Symposium on Recent Advances in Intrusion Detection* (*RAID2002*), LNCS, Vol.2516, pp.17–35 (2002).
10) Blum, A., Song, D. and Venkataraman, S.: Detection of Interactive Stepping Stones: Algorithms and Confidence Bounds, *Proc. 7th International Symposium on Recent Advances in Intrusion Detection* (*RAID2004*), LNCS, Vol.3224, pp.258–277 (2004).
11) Zhang, L., Persaud, A., Johnson, A. and Guan, Y.: Stepping Stone Attack Attribution in Non-Cooperative IP Networks, Technical Report 2005-02-1, Dept. of Electrical and Computer Engineering, Iowa State Univ. (2005).

12) Wang, X.: The Loop Fallacy and Serialization in Tracing Intrusion Connections through Stepping Stones, *Proc. 2004 ACM Symposium on Applied Computing* (*SAC'04*), pp.404–411 (2004).
13) Yung, K.H.: Detecting Long Connecting Chains of Interactive Terminal Sessions, *Proc. 5th International Symposium on Recent Advances in Intrusion Detection* (*RAID2002*), LNCS, Vol.2516, pp.1–16 (2002).
14) Yang, J. and Huang, S.-H.S.: A Real-Time Algorithm to Detect Long Connection Chains of Interactive Terminal Sessions, *Proc. 3rd International Conference on Information Security* (*InfoSecu'04*), pp.198–203 (2004).
15) Yang, J. and Huang, S.-H.S.: Matching TCP Packets and Its Application to the Detection of Long Connection Chains on the Internet, *Proc. 19th International Conference on Advanced Information Networking and Applications* (*AINA'05*), Vol.1, pp.1005–1010 (2005).
16) Peng, P., Ning, P., Reeves, D.S. and Wang, X.: Active Timing-Based Correlation of Perturbed Traffic Flows with Chaff Packets, *Proc. Second International Workshop on Security in Distributed Computing Systems* (*SDCS2005*), Vol.02, pp.107–113 (2005).
17) Postel, J.: TRANSMISSION CONTROL PROTOCOL, RFC 793, IETF (1981).
18) Braden, R.: Requirements for Internet Hosts—Communication Layers, RFC 1122, IETF (1989).
19) Ishiyama, M., Kunishi, M., Uehara, K., Esaki, H. and Teraoka, F.: LINA: A New Approach to Mobility Support in Wide Area Networks, *IEICE Trans. Communication*, Vol.E84-B, No.8, pp.2076–2086 (2001).

(平成 18 年 5 月 19 日受付)
(平成 18 年 11 月 2 日採録)

竹尾 大輔（正会員）

2004 年名城大学理工学部情報科学科卒業．2006 年同大学大学院理工学研究科情報科学専攻修了．同年三菱電機情報ネットワーク株式会社入社．ネットワークサービス事業部に所属．修士（工学）．2005 年マルチメディア，分散，協調とモバイル（DICOMO2005）シンポジウムヤングリサーチャ賞受賞．

伊藤 将志（学生会員）

2004 年名城大学理工学部情報科学科卒業．2006 年同大学大学院理工学研究科情報科学専攻修了．現在，同大学院理工学研究科電気電子・情報・材料工学専攻博士後期課程に在学中．VoIP，無線ネットワーク等の研究に従事．修士（工学）．電子情報通信学会所属．

鈴木 秀和（学生会員）

2004 年名城大学理工学部情報科学科卒業．2006 年同大学大学院理工学研究科情報科学専攻修了．現在，同大学院理工学研究科電気電子・情報・材料工学専攻博士後期課程に在学中．ネットワークセキュリティ，モバイルネットワーク等の研究に従事．修士（工学）．2006 年 IEEE 名古屋支部学生奨励賞受賞．2006 年マルチメディア，分散，協調とモバイル（DICOMO2006）シンポジウム松下賞（最優秀プレゼンテーション賞）受賞．電子情報通信学会所属．

岡崎 直宣（正会員）

1986 年東北大学工学部通信工学科卒業．1991 年同大学大学院工学研究科電気および通信工学専攻博士後期課程修了．同年三菱電機株式会社入社．2002 年宮崎大学工学部助教授．通信プロトコル設計，ネットワーク管理，ネットワークセキュリティ，モバイルネットワーク等の研究に従事．博士（工学）．電子情報通信学会，電気学会，IEEE 各会員．

渡邊 晃（正会員）

1974 年慶應義塾大学工学部電気工学科卒業．1976 年同大学大学院工学研究科修士課程修了．同年三菱電機株式会社入社後，LAN システムの開発・設計に従事．1991 年同社情報技術総合研究所に移籍し，ルータ，ネットワークセキュリティ等の研究に従事．2002 年名城大学理工学部教授，現在に至る．博士（工学）．電子情報通信学会，IEEE 各会員．