SCIS 2006 The 2006 Symposium on Cryptography and Information Security
Hiroshima, Japan, Jan. 17-20, 2006
The Institute of Electronics, Information and Communication Engineers

コネクション検出方式による踏み台攻撃検出手法の提案

A Proposal of a Detection Technique on Stepping-Stone Attacks using Connection Detection Method

竹尾 大輔* 岡崎 直宣† 渡邊 晃*
Daisuke Takeo Naonobu Okazaki Akira Watanabe

あらまし 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共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，検出にある程度の時間が必要である．本稿では，踏み台攻撃が行われるとき，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必ず送信されることに着目し，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検出方式を提案する．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，踏み台攻撃を確実に検出できることを確認した．

キーワード ネットワークセキュリティ，侵入検知，踏み台攻撃

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，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，常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取られている．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すると（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ず非常に危険である．踏み台攻撃は，踏み台ホストに対しては必ずリモートログインでアクセスする．踏み台ホストから被害ホストへは，対話型で攻撃ができるリモートログインを用いる場合が多いが，それ以外のあらゆるアクセス方法がありうる．管理者は管理目的のために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アクセス自体は正常な行為であり，上記のような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．

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．いずれも踏み台ホストから被害ホストまで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．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は，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のデータマッチング[1]や，コネクションへの透かし埋め込み[2]による検出を行っている．この方式は検出にパケットの中身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[3]～[10]はリモートログイン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

* 名城大学大学院理工学研究科，〒468-8502 愛知県名古屋市天白区塩釜口 1-501, Graduate School of Science and Technology, Meijo University, 1-501, Shiogamaguchi, Tenpaku-ku, Nagoya-shi, Aichi 468-8502, Japan.
† 宮崎大学工学部，〒889-2192 宮崎県宮崎市学園木花台西 1-1, Faculty of Engineering, University of Miyazaki, 1-1, Gakuenkibanadai-Nishi, Miyazaki-shi, Miyazaki 889-2192, Japan.

用した研究が主流であり，検出精度を高めるために様々な提案がなされている．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もので，被害ホストへのアクセスが FTP のようなリモートログイン以外の場合は検出することができない. また，タイミングベース方式は，相関関係を見る関係上ある程度の検出時間を必要とするという課題がある．

本稿では，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 TCP コネクション確立要求が送信されることを検出するコネクション検出方式を提案する．この手法は，被害ホストが提供している TCP サービスを起動するコマンドを乗せたリモートログインパケット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．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得る．

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．そして最後に 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, Foothold，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．Foothold（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 Foothold 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 Foothold 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 Foothold を経由した攻撃もありうるが，本論文では Foothold が 1 台の場合について着目する．Attacker から Target までの間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，Foothold を経由して Target にアクセスする手段を踏み台攻撃と呼ぶ．Attacker から Foothold へのアクセスのフローを Flow X, Foothold から Target へのアクセスのフローを Flow Y と呼ぶ．リモートログインプロトコルとしては，SSH，Telnet，Rlogin がある．

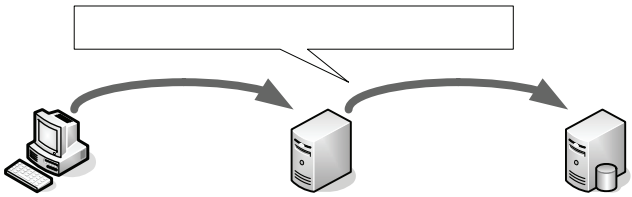

図 1　踏み台攻撃モデル

2.2　タイミングベース方式

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，両 Flow の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が，パケットが暗号化されると適用できないため，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相関関係があることを検出する．図 2 のように，ある期間内に Foothold 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間隔などを手がかりにし，2 つのフローに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．

しかし，この手法は Target へのアクセスにリモートログイン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．

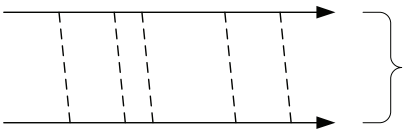

図 2　相関関係の発見

3　提案方式

Attacker が Foothold へとリモートログインした後，そこから Target に対してリモートログインを含めたあらゆる TCP 通信でアクセスする可能性を想定し，これを第三者的にネットワーク上で検出する方式を提案する．この方式をコネクション検出方式と呼ぶ．踏み台攻撃の検出はネットワーク上の監視ホスト（以降，Detector と

呼ぶ）で検出する．Detector は組織ネットワークのゲートウェイの直下のように，ネットワークの外部と内部の間を流れる通信を監視できる位置に設置する．

図 3 にコネクション検出方式の原理を示す．まず Attacker が Foothold 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 Foothold 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，Foothold 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．リモートログイン通信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．

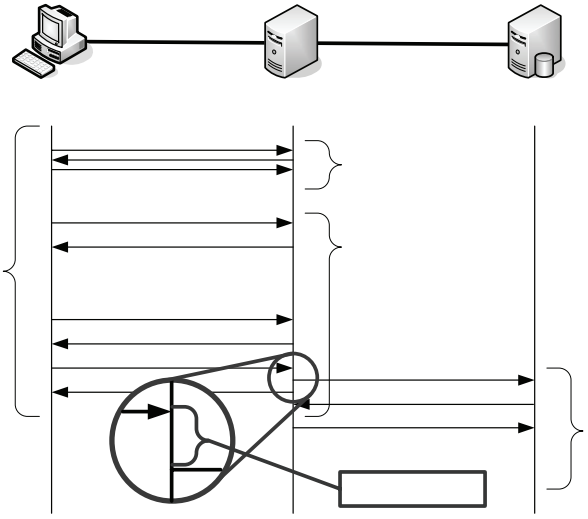

図 3　コネクション検出方式の原理

以上の原理に従い，コネクション検出方式では 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)　リモートログイン通信パケットの監視

TCP パケットのうち，SSH，Telnet，Rlogin などのリモートログインの通信パケットでかつ PUSH フラグのセットされたパケットを検出する．検出する度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．PUSH フラグを見る理由は，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，PUSH フラグが必ずセットされているからである．

(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，リモートログイン通信パケットの宛先 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，リモートログイン通信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，踏み台攻撃と判断する．SYN フラグを見る理由は，Attacker からのコマンドを受けて Foothold 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．

コネクション検出方式ではリモートログイン通信パケットのデータ内容を参照する必要が無いので，リモートアクセスが暗号化されていれもかまわない．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，Foothold 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

コネクション検出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，Detector が Foothold の通信パケットを監視できるようにした．プログラムのモジュール構成図を図 4 に示す．

リモートログイン受信記録はリングバッファとして実装した．TCP コントロールフラグフィールドに PUSH フラグがセットされたリモートログイン通信パケットを受信するごとに，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，受信時刻の 5 つのフィールドから成るレコードを記録していく．

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ので処理内容は極めて単純である．

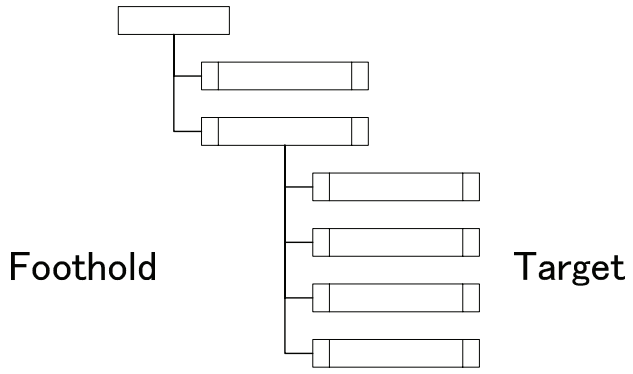

図 4　プログラムの構成

5　評価

5.1　評価環境

評価環境を図 5 に示す．踏み台攻撃モデルを構成する Attacker，Foothold，Target に加えて，提案方式を実装した Detector を用意した．これらの機器を 100BASE-TX の LAN によって接続した．Foothold では SSH，Telnet，Rlogin，FTP サービスを，Target で

は Telnet，FTP サービスを起動させた．Detector，Foothold のマシンスペックは表 1 の通りである．この評価環境において実際に踏み台攻撃を行い，Detector で検出できることを確認した．Attacker から Foothold へのリモートログインには SSH，Telnet，Rlogin を，Foothold から Target への TCP 通信には Telnet，FTP を用い，いずれのリモートログインにおいても踏み台攻撃を検出できることを確認した．

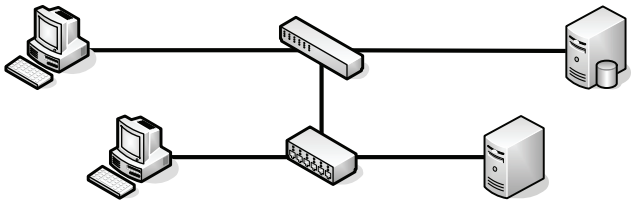

図 5　評価環境

表 1　Detector，Foothold のマシンスペック

	Detector	Foothold
CPU	PentiumIII 800MHz	Pentium4 2.4GHz
RAM	256MB	256MB
OS	FreeBSD 5.2.1-RELEASE	FreeBSD 5.3-RELEASE

5.2　踏み台検出時間測定結果

リモートログイン SSH パケットを検出してから TCP サービスの SYN パケットを検出するまでの時間（以降，踏み台検出時間と呼ぶ）を背景負荷が無い状態で測定した結果を表 2 に示す．表 2 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，全く同じ 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示すためであり，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あることを利用している．

表 2　踏み台検出時間測定結果

		Flow Y			
		Telnet		FTP	
		①	②	①	②
Flow X	SSH	36.38	30.47	30.80	25.82
	Telnet	36.32	30.69	30.56	25.66
	Rlogin	36.17	30.62	30.53	25.65

※表中の①は起動時接続先指定，②は起動後接続先指定の意味
※値は 10 回試行の平均（ミリ秒）

リモートログインによって若干異なるが，Telnet 起動時接続先指定では 36 ミリ秒程度，Telnet 起動後接続先指定では 30 ミリ秒程度の間に SYN パケットが送信されていることがわかる．また FTP 起動時接続先指定では 30 ミリ秒程度，FTP 起動後接続先指定では 25 ミリ秒程度となり，Telnet より短い結果が得られた．

踏み台検出時間が変動する要因を調査するために，Foothold に対して FTP 接続および HTTP 接続をそれぞれ行い，ファイル転送により Foothold に背景負荷を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 のファイル転送数を 0 から 10 へ増加させていき，Flow X と Flow Y がともに Telnet である場合の踏み台検出時間の測定を行った．

図 6 に FTP を背景負荷としてファイル転送数を変えた場合の踏み台検出時間を，図 7 に HTTP を背景負荷としてファイル転送数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するにつれて踏み台検出時間も比例に近い関係で増加していることが分かる．背景負荷が FTP と HTTP のどちらのときであっても踏み台検出時間は同じであった．

踏み台検出時間の変化に影響を与えたものが，CPU の処理負荷なのかネットワークインタフェースの処理負荷なのかを明らかにするために，Foothold に背景負荷を与えたときと同一の条件において，Foothold の CPU 全体の負荷状態，FTP デーモンおよび HTTP デーモンの 1 プロセス当たりが与える CPU 負荷がどの程度になるかを測定した（図 8，図 9）．CPU 全体の負荷としては，FTP を背景負荷とした場合，セッション数が 2 つ以上になると 85～90%を占めていることが分かる．HTTP を背景負荷とした場合，セッション数が 2 つ以上になっても 25%前後で済んでおり，それ以上には増加していないことが分かる．これに対して，FTP デーモンの 1 プロセス当たりが与える CPU 負荷としては，背景負荷が FTP と HTTP のどちらであってもセッション数が増加するにつれて減少していく．また，背景負荷が FTP の場合に比べて HTTP の場合は CPU 負荷が遥かに小さいことが分かる．図 8，図 9 には各ファイル転送数における FTP デーモンおよび HTTP デーモンが与える CPU 負荷の合計も示した．複数起動している FTP デーモン全体が与える CPU 負荷は 55～65%で推移しているが，HTTP デーモンの場合は合計しても 5%以下である．背景負荷が HTTP の場合の CPU 負荷が 25%前後で済んでいるのは，HTTP デーモンに割り当てられる CPU 処理能力の合計が小さいことが理由として考えられる．

以上のことから，ファイル転送数と CPU 負荷との間に相関は見られず，踏み台検出時間は背景負荷のファイル転送のセッション数，すなわちネットワークインタフェース処理負荷に大きく依存していることが分かる．

Foothold に対する通信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（フォールスネガティブの増加）．逆に監視時間を長くすれが必ず検出できるが，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の PUSH 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 を特定しづらくなる可能性がある（フォールスポジティブの増加）．この課題の対策案としては，Foothold に対する通信量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が考えられる．また，踏み台検出時間はマシンスペックによっても変化する可能性があり，これらの状況に合わせて監視時間を設定する必要がある．

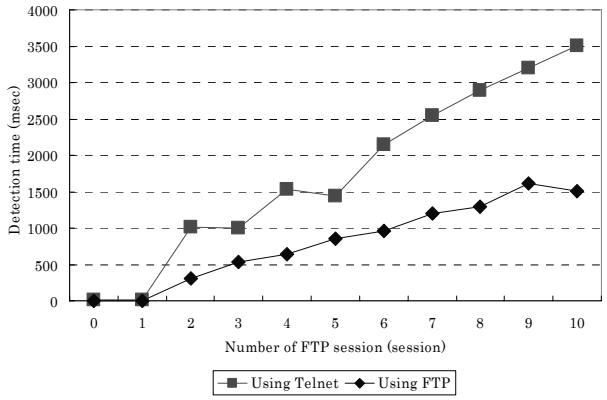

図 6　FTP 背景負荷がある時の踏み台検出時間

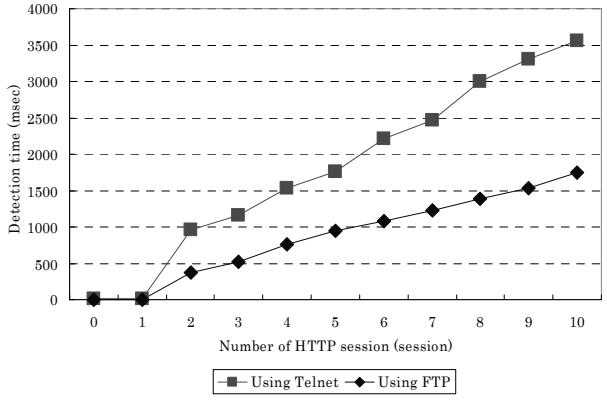

図 7　HTTP 背景負荷がある時の踏み台検出時間

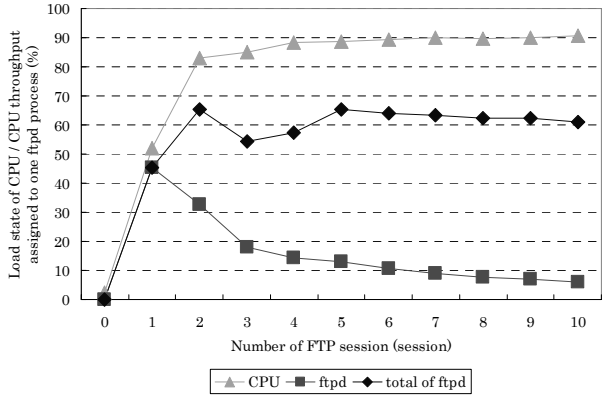

図 8　FTP 背景負荷がある時の CPU 状態

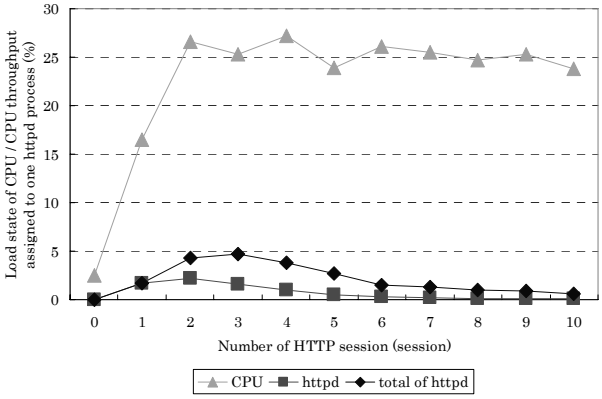

図 9　HTTP 背景負荷がある時の CPU 状態

5.3　今後の課題

コネクション検出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のみを手がかりとしているため，フォールスポジティブが増加しやすく，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．また，リアルタイム性を持たせつつ検出精度を向上させる手法についても検討していく必要がある．

なお，本研究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ていない．即ち，Target に対して，UDP による攻撃を行う場合がありうる．この場合は Target 上で UDP によるサービスを起動していることが条件となるが，重要なサーバ上ではこのようなサービスは起動しないことが望まれる．次に，事前に Foothold 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，例えば 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．また，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．このような攻撃はリアルタイムで Target の状況を把握できないので攻撃には高度な技術が必要となる．今回の方式ではこのように Foothold 上で大きな時間差のある攻撃を検出することは不可能であり，別の手段を適用する必要がある．例えばリモートコントロールプログラムを Foothold 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，事前に攻撃を防ぐことができると考えられる．

踏み台は正規のユーザが正規の手段として行う場合も考えられる．悪意のある攻撃者が行う踏み台攻撃を検出するためには，本方式に加えて踏み台の正常・不正の判断をしなければならない．事前に正常・不正パターンを定義した IP アドレスリストなどを用意することで，踏み台の正常・不正の判断を行う必要があると考えられる．

6　まとめ

本稿では, Foothold に対するリモートログイン操作の終了と同期して Foothold から Target に対して TCP コネクション確立要求が送信されることを検出する踏み台攻撃検出方式について提案した．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を取っているため, Foothold に対するリモートアクセス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く，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．しかし，Foothold 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ため，適切な監視時間の設定が必要であることが分かった．今後は，検出精度を高める方法や，踏み台の正常，不正を判断し，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] S. Staniford-Chen and L. T. Heberlein, “ Holding Intruders Accountable on the Internet, ” 1995 IEEE Symposium on Security and Privacy, pp.39-49, May 1995.

[2] X. Y. Wang, D. S. Reeves, S. F. Wu and J. Yuill, “ Sleepy Watermark Tracing: An Active Intrusion Response Framework, ” the 16th International Information Security Conference (IFIP/Sec'01), June 2001.

[3] Yin Zhang and Vern Paxson, “ Detecting Stepping Stones, ” 9th USENIX Security Symposium, pp.171-184, Aug. 2000.

[4] Kunikazu Yoda and Hiroaki Etoh, “ Finding a Connection Chain for Tracing Intruders, ” 6th European Symposium on Research in Computer Security (ESORICS 2000), pp.191-205, Oct. 2000.

[5] X. Wang, D. S. Reeves and S. F. Wu, “ Inter-packet delay based correlation for tracing encrypted connections through stepping stones, ” 7th European Symposium on Research in Computer Security (ESORICS 2002), pp.244-263, Oct. 2002.

[6] Xinyuan Wang and Douglas S. Reeves, “ Robust Correlation of Encrypted Attack Traffic through Stepping Stones by Manipulation of Interpacket Delays, ” 10th ACM conference on Computer and communications security (CCS 2003), pp.20-29, Oct. 2003.

[7] W. T. Strayer, C. E. Jones, I. Castineyra, J. B. Levin and R. R. Hain, “ An integrated architecture for attack attribution, ” BBN Technologies, Tech. Rep. BBN REPORT-8384, Dec. 2003.

[8] David L. Donoho, Ana Georgina Flesia, Umesh Shankar, Vern Paxson, Jason Coit and Stuart Staniford, “ Multiscale Stepping-Stone Detection: Detecting Pairs of Jittered Interactive Streams by Exploiting Maximum Tolerable Delay, ” 5th International Symposium on Recent Advances in Intrusion Detection (RAID 2002) LNCS-2516, pp.17-35, Oct. 2002.

[9] Avrim Blum, Dawn Xiaodong Song and Shobha Venkataraman, “ Detection of Interactive Stepping Stones: Algorithms and Confidence Bounds, ” 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004), pp.258-277, Sept. 2004.

[10] Linfeng Zhang, Anthony G. Persaud, Alan Johson and Yong Guan,“ Stepping Stone Attack Attribution in Non-Cooperative IP Networks, ” Technical Report 2005-02-1, Department of Electrical and Computer Engineering, Iowa State University, 2005.