ネットワークセキュリティにおける インサイダー脅威対策

研究代表者　　堀　良彰　　九州大学大学院システム情報科学研究院・准教授

1 はじめに

ネットワークセキュリティにおけるインサイダー脅威対策として、インサイダー脅威対策のための枠踏みに関する議論およびHTTPリクエストを用いた情報漏えい検知手法について研究を行った。

2 ネットワークセキュリティにおけるインサイダー脅威対策

2-1 ネットワークセキュリティにおけるインサイダー脅威対策の枠組み

（1）はじめに

インサイダー脅威とその対策について問題となっている。インサイダー脅威は、権限を有するものの裏切りなど、内部と外部を区別する境界が存在しないことから、情報セキュリティ対策として従来から行われてきた境界におけるアクセス制御は、そのポイントが明確でなくなるため困難となる。

インターネットのようなパケット交換アーキテクチャを有するネットワークにおけるアクセス制御対策は、セキュリティポリシが異なる境界にゲートウェイやファイアウォールを設置し、セキュリティポリシに基づくルールによりパケットの通過または遮断を制御するものであった。ネットワークアクセスは、ファイアウォールと呼ばれるゲートウェイを通過するパケットによって行い、外部からのアクセスを内部に許したくない場合は、始点アドレスが外部ネットワークのもの終点アドレスが内部ネットワークのものであるパケットを遮断する方法で行われた。ゲートウェイを通過するパケットに対して高度な分析を行い攻撃を認識した場合にそれを遮断する侵入防御システム（IPS: Intrusion Prevension System）も実現されているが、セキュリティポリシが異なる境界におかれる点は同じである。

従来の外部脅威に対するネットワークにおけるアクセス制御と比較して、インサイダー脅威においては、インサイダー（内部のもの）が自身が有する権限をセキュリティポリシに反するように乱用することから、ネットワークにおけるアクセス制御ルールが複雑になる問題がある。さらに、従来の外部脅威対策のためのファイアウォール設置のように、外部と内部の境界が明確になっているわけではないため、どこでアクセス制御のためのポイントを設けるかが問題となる。

本稿では、これらの問題への解決のために、ネットワークにおけるインサイダー脅威対策として、（1）異常検知に基づくアクセス制御ルールの動的生成と、（2）動的に生成されるアクセス制御機構設置について議論を行う。

（2）インサイダー脅威とその対策関連研究

ネットワーク上の攻撃は外部からだけとは限らない。システム内部からの攻撃は、予測困難でありその影響も深刻であることは1980年代から認識されていた。2000年代には、金融系サービスにおける内部脅威に関する報告書がまとめられ、また内部脅威の定義を試みる論文も登場してきた。2008年には、M.Bishop やD,Gollmann など欧米の著名なコンピュータセキュリティ研究者が、伝統あるDagstuhl セミナーで「内部脅威対策」(http://www.dagstuhl.de/08302)を討議し、2010年夏には継続したDagstuhlセミナー「内部脅威：回避・緩和・対応戦略」(http://www.dagstuhl.de/10341)を開催している。昨年からは、内部攻撃脅威に特化した国際会議 International Workshop on Managing Insider Security Threats (2009, 2010) ならびに1st ACM CCS workshop Insider Threat 2010 が発足し研究コミュニティ形成が勧められている。学術雑誌に関しては、Security and Communication Networks (SCN)2010ジャーナルにおいて、特集 “Defending Against Insider Threats and Internal Data Leakage” などが企画され、欧米を中心に急速に学術研究コミュニティが形成されつつある。

Matt Bishopは､インサイダーとインサイダー脅威について次のよう述べている。“an insider is a trusted entity that is given the power to violate one or more rules in a given security policy. Enforcement mecha- nisms are not applied against those trusted users. The insider threat occurs when a trusted entity abuses that power.” これは、インサイダー脅威を論じるための次の重要な視点を与えている。こ

れらのことから、インサイダー脅威対策においては、情報資源に対するアクセスが本来あるべき以上に過剰となっている挙動について見つけ出す必要がある。

・インサイダーはセキュリティポリシに基づくルールに背くことを可能にする力や権限を有する信頼されたユーザ等（エンティティ）である。
・インサイダーは、（前提としては、信頼されているため）強制手段についてその振舞に制約をかけられているわけではない。
・信頼されているユーザ等が、その力や権限を乱用する場合に、インサイダー脅威生じる。

（３）インサイダー脅威対策

これまでのネットワークにおける外部脅威対策においては、脅威と守るべき情報資源の境界が物理的にはっきりしており、そこにファイアウォール等のアクセス制御機構を設置して脅威に対応してきた。ところが、インサイダー脅威においては、権限を有するものの裏切りが問題となるなどアクセス制御を行うための外部と内部の境界が最初から存在しない。さらに、従来の脅威対策では、自組織の外部から、自組織の情報資源を保護するためのセキュリティポリシを事前に作成し、それから導出されるネットワークアクセス制御ルールをファイアウォールや侵入検知システムに事前に備え、それらのルールと照らし合わせることにより外部脅威への対策を行ってきたが、インサイダー脅威についてはこのような対策をとることができない。

前節で述べたように、インサイダー脅威対策においては、予め許可された者による情報資源に対するアクセスが本来あるべき以上に過剰となっている挙動を発見する仕組みを構築する必要がある。この考えは、従来の侵入検知における不正検知（misuse detection）や異常検知（anomaly detection）に相当する。さらに、これらの不正検知ならびに異常検知は、アクセス過剰だけでなく、アクセス不足を検知することは、本来必要なアクセスがなされていないことを発見する手立てにもなり得る。

インサイダー脅威対策としての不正検知

不正検知は、「不正（misuse）」を示すルールを定め、それとの比較により検知を行う。つまり、予め不正を形式化しておく必要がある。インサイダー対策においては、不正の形式化自体をどのように行うかということ自体問題となる。信頼されているユーザが取り得る振舞を示す空間が存在すると、不正な振舞あるいは正当な振舞を形式化する必要がある。そのためには、不正の分類が必要とされる。既存研究では、比較的単純なものについてネットワークアクセスに関する不正の分類を行っているが、このような分類を行うにはシステムに関する高度な知識が必要となる。

インサイダー脅威対策としての異常検知

信頼されているユーザが取り得るべき振舞い、つまり正当な振舞いを決定的に判別するための形式的な定義が簡単でない場合には次の方法を検討できる。実際のシステムにおける振舞い挙動を基に学習理論を用いて生成した通常の振舞いモデルを「正当な振舞い」に類似するものとみなし、それに合致しない異常を検知することで、不正検知にかかる工数を削減することにつながる。

インサイダー脅威対策のポイント

従来のネットワーク脅威対策においては、脅威と守るべき情報資源の境界が物理的もしくは論理的で明確である場合、その箇所に、そこにファイアウォール等のアクセス制御機構を設置して脅威に対応してきた。ところが、インサイダー脅威においては、権限を有するものの裏切りなど境界が最初から存在しない。そこで、脅威から情報資源を守るためのアクセス制御機構の配置場所についての議論が必要となる。そのためには、アクセス対象となる情報資源と、それをアクセスするユーザの位置を明確にし、その間で必要となるアクセス制御を行う必要がある。特に、異常検知ベースの対策手法をとる場合には、適切なポイントにアクセス制御ルールを動的に追加する機構が必要となる。

（４）インサイダー脅威対策の枠組み

前節での議論から、ネットワークにおけるインサイダー脅威対策のためには、情報資源にアクセスするユーザ（信頼されているアクセス者）がどのようにアクセスしようとしているかの振舞いに関する情報を通信チャネル上で伝送される情報を用いてモニタ記録し、それを分析することで、アクセスが正常に行われているか不正に行われているかを判定する必要がある。もし、振舞に異常が見られた場合は、アラームを上げることにより、不正検知にかかる工数を減少させることができる。図１は、インサイダー脅威に対応するための枠組みを示す。

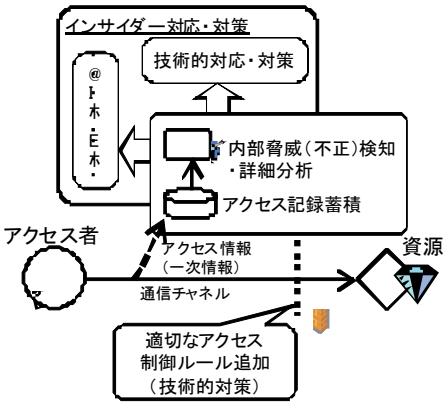

図１ インサイダー脅威に対応するための枠組み

内部脅威解析者は、情報資源へのアクセス状況を蓄積しログ収集に努めるとともに、逐次異常がないか解析を行う。解析の結果、異常が判別された場合には、それがポリシに反していないか比較注意する。このような異常検知を用いてもインサイダー脅威対策を行うことができる。

過去のアクセス等の履歴情報を含むアクセス情報から、通常業務とは異なる振舞を検知する機構を実現することでインサイダー脅威に対抗することができる。そのために、機械学習分野において研究がすすめられている異常検知技術を導入し、アクセス者の挙動に関してそのモデル化ならびに異常検出を実施する。本研究では、時間の変化に伴いネットワークを介してアクセスを行う利用者の挙動を、一連の挙動としてモデル化することで、ネットワークを用いた異常アクセスを検出する手法を導入する。情報資源のアクセス主体からのネットワークを介した情報システムへのアクセスを学習により正常を示すルールおよび異常を示すルールとして、ルール生成を行う。生成されたルールとの対比により異常アクセスを検知する。

（５）動的アクセス制御について

これまでの脅威対策においては、脅威と守るべき情報資源の境界が物理的にはっきりしており、そこにファイアウォール等のアクセス制御機構を設置して脅威に対応してきた。ところが、インサイダー脅威においては、権限を有するものの裏切りなど境界が最初から存在しない。そこで、脅威から情報資源を守るためのアクセス制御機構を動的に生成し、配置する必要がある。

近年のコンピュータハードウェア資源をネットワークで十分高速に接続されたデータセンターに集中させ効率的に利用するクラウドコンピューティング技術が普及しつつある。クラウドコンピューティング環境においては、計算機資源やネットワークが仮想化されるために、従来と比較してどこでどのようにネットワークのアクセス制御を行ってよいのかポイントを示せない状況にある。そこで、インサイダー脅威を確認した場合、必要なフィルタリングを動的に開始できるようなシステムを構築する必要がある。

（６）おわりに

インサイダー脅威とその対策について議論を行った。インサイダー脅威は、権限を有するものの裏切りなど、内部と外部を区別する境界が存在しないことから、情報セキュリティ対策として従来から行われてきた境界におけるアクセス制御は、そのポイントが明確でなくなるため困難となる。本研究調査では、インサイダー脅威検知のため、異常検知手法をインサイダー脅威のモデルに適用し、通常業務とは異なる振舞を機械的に評価する手法の考案を行う。特に、ネットワークサービスへのアクセス状況をモニタし、異常なアクセスを数理的手法により評価する。検知されたインサイダー脅威を防止するために、脅威から情報資源を守るためのアクセス制御機構を動的に生成し、配置する必要がある。これらの研究を実施することにより、ネットワークセキュリティの観点かインサイダー脅威に対抗するための技術の確立を目指す。

2-2 HTTP リクエストを用いた情報漏えい検知手法

（１）はじめに

情報の漏洩問題に直面する機会が増えてきている。漏洩経路は様々であるが，中でもインターネットによる漏洩被害者数はその割合の多くを占めている。その漏洩に対しては技術的な対策が必要であるが、十分ではないのが現状である。

インターネットを介した漏洩原因にスパイウェアがある。スパイウェアとは個人情報等を盗み出す悪性ソフトウェアである。住所や氏名、クレジットカード番号やパスワード等の個人情報を不正に取得するスパイウェアは増加している。初期のスパイウェアは URL 履歴などの情報を無造作に収集するものが主であったが、近年では PC 内の重要な個人情報を盗み出すスパイウェアが増えている。近年普及しつつあるスマートフォンの Android OS でも同様の漏洩が確認されている。Android OS 対応の広告付きアプリなどが、個人情報を送信することによって漏洩が発生する。Paul らの調査によると、Android マーケットの 49%のアプリに何らかの広告ライブラリが入っており、その中の 56%が本来は必要が無いと思われるパーミッションを得ている。パーミッションを得たアプリは位置情報などを使用者の意図しないところで製作者へ送信する恐れがある。

スパイウェアには収集した情報を HTTP リクエストに記載し,攻撃者のサーバに送信することで漏洩を引き起こす。近年では多くの Web アプリケーションが HTTP 通信を行なっており、ポート番号によるパケットフィルタで遮断することは現実的ではない。そこで本研究では HTTP リクエストを用いた情報漏洩を検知するシステムを提案する。その手法として、既存研究の漏洩情報量の数値化手法を検討したものを使用する。数値化手法を使用することで、通常のリクエストサイズを求める場合と比べ、漏洩情報が含まれている際に生じる通信量の外れ値の発見が容易になる。また、数値化手法を使った際の検知の精度を調査するために漏洩検知の模擬実験を行った。今回実験対象としたのは PC 上でブラウザを使って Web ページを閲覧する際に送信される HTTP リクエストである。しかし、通信量の外れ値を観測するという点では Web アプリケーションや Android

端末を使用する際に送られるHTTPリクエストにおいても共通しているため、本手法は同様に適用できると考えられる。模擬実験では漏洩情報量が1000バイトの時に挙げられたアラートの誤検知率は11.6%となり、誤検知の少ないアラートが挙げられることが確認できた。

（２）関連研究

Borders、Prakashらは HTTP リクエストにどれだけ新しい情報が含まれているかを数値化することを提案している。数値化の手法として、送信する最新の HTTP リクエストと直前の HTTP リクエストとの編集距離を求めている。編集距離とは文字列をある文字列に一文字ずつ変換するのに必要な手順の回数である。今回はこの手法により、既に送られているデータはカウントされないことになる。例を挙げると、com と co.jp の編集距離は、置換1回、挿入2回と計3回の操作で文字列変換を完了しているので編集距離は3となる。Bordersらは編集距離を求める時間を短縮する方法について述べている。編集距離を通常通り求めると計算量は$O(n^2)$となるが、文字列を分割してそれぞれの編集距離を求めることでこれを短縮することができる。例えば4文字と4文字の文字列を比較するよりも、4文字を半分に分割して2文字ごとに分けて比較するほうが計算量が高速となる.

なおHTTPリクエストにおけるHost、Referer、Cookieフィールドでは単純に編集距離を求めずに別の手法をとっている。これは他のフィールドがあまり変化しないのに対して、これらのフィールドは頻繁に変化する性質があるためである。Hostフィールドは接続するサーバを示す。直前に閲覧していたページの HTML に存在するURLのホスト部分と一致するものがあれば、Hostフィールドはカウントしない。そうでない場合は文字数をそのままカウントする。

Refererフィールドは直前に見ていたページのURLを示す。直前のHTTPリクエストがRefererフィールドのURLを含んでいた場合はカウントしない。そうでない場合は文字数をそのままカウントする。Cookieフィールドは Web ブラウザに保存された情報を示す。サーバの HTTP レスポンスによって生成され、認証等に用いられる。HTTP レスポンスの内容から Cookie の内容を想定し、想定した Cookie が送信されていればカウントしない。想定していない Cookie であれば想定した内容との編集距離を求める。

以上のような数値化手法をとることで、古い情報を無視してカウントすることができる。特定のブログを巡回した際の HTTP リクエストを収集し、それを数値化した際の平均は 1.9 バイトであった。これは実際のHTTPリクエストの平均サイズの0.32%にあたる。

この数値化手法では、直前のHTTPリクエストとのみ比較をして編集距離を求めている。しかしながら、2つ前や3つ前のHTTPリクエストに同じ情報が存在する場合に、それを古い情報とみなすことができない。

また, Borders は Web Tap Personal beta 1.2 と呼ばれるスパイウェア検知システムも実装している. このシステムではスパイウェアによる異常通信の検知を目的としているので、自らの意志を持って情報を発信した際の漏洩は対象としていないと思われる。その動作を調査したところ、掲示板の書き込みに対して使われるPOSTリクエストによるメッセージ送信の検出は行われていなかった。

（３）情報量数値化手法

近似情報量の算出とその利点

HTTPを利用した編集距離の算出に基づく、既存の情報漏洩検知手法よりも古い情報を無視するため、履歴情報を用いた情報量近似手法を提案する。この手法によって数値化されたサイズを本稿では近似情報量と呼ぶ。なお、その単位はバイトとして扱うことにする。情報漏洩を検知する手法としてHTTPリクエストサイズを測定するのみでは、リクエストサイズに対して漏洩情報が小さすぎるため、発見が困難である。それに対して近似情報量を測定する場合は、繰り返される情報は無視されるため全体的に小さな値が観測される。漏洩情報は繰り返される情報ではないため、この手法では近似情報量が際立つことになり発見が容易になる。関連研究では直前のHTTPリクエストとのみ比較をしていたが、本研究では直前だけではなく、さらに以前のHTTPリクエストとの編集距離を求めることで、より古い情報を無視することを試みる。

特定フィールドにおける処理

次にHost、Referer、Cookieフィールドについて示す。Hostフィールドにはドメイン名が入るので、これをDNSサーバに問い合わせる。正常な返答が返ってきた場合は、フィールドの書き換えによる漏洩はないと判断できる。その場合は近似情報量は0バイトとする。そうでない場合は編集距離を求める。

Refererフィールドについては送信するHTTPリクエストよりも前のHTTPリクエストを参照し、URLを生成し比較する。Cookieフィールドについては以前に送ったCookieフィールドを保存しておき、そのいずれかと一致するものがあれば近似情報量を0バイトとする。一致するものがない場合は編集距離を求める。これは一度送ったCookieフィールドはほとんど変化することはないという性質を利用したものである。

リクエストURIについてはHTTPレスポンスに含まれるHTMLの内容からURL部分を抽出し比較する。一致

するものがなかった場合は編集距離を求める。URL には HTML に直接書かれる静的なものと、Javascript 等によって動的に生成されるものがある。静的なものは正規表現による抽出が容易であるが、動的なものは抽出が容易でない。その動的な URL 抽出方法について関連研究では議論している。今回対象としたのは HTML に直接記載されている静的な URL のみであり、関連研究のように動的な URL を抽出するのは今後の課題である。

（４）数値化手法の評価

提案する数値化手法の評価実験を行った。本稿の実験では 2011 年 4 月に著者が収集したキャンパス LAN のトラフィックを用いた。この LAN トラフィックは，ある大学の研究室メンバーのインターネットを用いた日常の Web ブラウジングにより生成されたものである。tcpdump により収集した pcap ファイルに数値化プログラムを実行することで数値化を行う。実験環境を次に示す。

・OS: Fedora 14
・CPU: AMD Turion(tm) Neo X2 Dual Core Processor L625 1.60GHz
・メモリ: 512MB

まず、履歴参照数と算出される数値の間の関係を調べるために実験を行った。実験対象は単一 IP アドレスから送信される HTTP リクエスト 500 個である。なお IP アドレスは無作為に選んだものであり、HTTP リクエストは PC 上でブラウザを利用する際に送られるものである。この実験では,履歴参照数は 1 から 19 とした。履歴参照数と近似情報量との関係を調査した。履歴参照数を増やすことで数値が低くなっていることがわかる。また、直前だけではなくさらに前の HTTP リクエストまでさかのぼって比較をすることで、古い情報をより無視することができたと考察する。

また、履歴参照数と本プログラムの実行時間の関係を調査する。履歴参照数が増加するのに比例して実行時間も増加している。これより精度の向上と実行時間の減少を両方満たすことは困難であるといえる。したがって、履歴参照数はそれぞれの増減率・減少率等から判断して決定する必要がある。

次に，本手法を用いることでどの程度の情報が無視できているのかを実験する。まず HTTP リクエストの文字数をそのままカウントし、HTTP リクエストのサイズを調べる。実験対象は単一 IP アドレスから送信される HTTP リクエスト 11259 個である。前実験と同様に、IP アドレスは無作為に選び、HTTP リクエストは PC 上でブラウザを利用する際に送られるものである。各バイト数の HTTP リクエストの分布を調査した。リクエストサイズは 0 バイトから 3000 バイト付近までの範囲に散らばっており、この時の平均サイズは 940 バイトであった.

次に本プログラムの比較手法を用いて数値を観測する．設定する履歴参照数は 10 とする．各近似情報量の出現頻度の分布を調査した。その結果、近似情報量が 0 バイトから 400 バイト付近の小さい範囲に集中していることがわかる。提案手法により多くの古い情報を無視できた結果、HTTP リクエストが持つ新しい情報の量に近い値が算出できたといえる。なお、今回の実験では単一 IP アドレスを対象にしている。例えば NAT などが利用され、多数の端末が同一 IP アドレスに集約される場合はその IP アドレスは複数の端末で共用され、混在するアプリケーションプロセスが生成する HTTP リクエストが送信されることになる。結果的に編集距離の測定による提案手法では大きな値が算出されることとなる。したがって、環境によってはグローバル IP アドレスではなくローカル IP アドレス毎等、アプリケーションを識別する手法を適用した上で近似情報量を算出することが必要になると考えられる。

（５）検知システムへの応用

本研究で提案する HTTP リクエストにより伝送される情報量数値化手法の応用として、情報漏洩検知システムを提案する。提案するシステムではリアルタイムでトラフィックを監視し、近似情報量を算出する。特定のサイトを継続して閲覧している場合では、内容の類似した HTTP リクエストが継続して送信されるので近似情報量は低い数値を示す。新しい情報が大量に送信されている HTTP リクエストでは高い数値を示す。その HTTP リクエストは利用者が普段送信しない異常なものと判断できるのでアラートを挙げる。例えば利用者が掲示板等に書き込む際、POST メソッドを利用して HTTP リクエストを送信した場合は高い数値が算出される。

これらを踏まえて検知システムを設計する。このシステムでは観測された数値があるしきい値を超えていた場合にアラートを挙げる。このシステムがどの程度の検知ができるのかを評価するために、漏洩が起きていることを想定して次のような模擬実験を行う。

・無作為に選んだメンバー一人の,PC 上での Web ブラウジングによって送信された HTTP リクエストを 1069 個収集する。
・その 1 割の 107 個の HTTP リクエストのフィールドに漏洩情報としてランダムな文字列（30 バイト，1000 バイト）を付加する。

・システムで HTTP リクエストを解析（履歴参照数は 0、1、10）し、誤検知率を算出する。

しきい値は未検知数が 0 かつ誤検知数が最も少なくなるように設定する。具体的には漏洩情報量が 30 バイトの時は 29 バイト、漏洩情報量が 1000 バイトの時は 780 バイトである。検知数と誤検知数から算出した誤検知率を調査する。

その結果、履歴参照数を増やすことで誤検知率が下がっていることが確認できる。また、漏洩情報量が 1000 バイトと大きい時は誤検知率の比較的小さなアラートが挙げられていることが確認できる。それに対して、漏洩情報量が 30 バイトと小さい時は誤検知率が大きい結果となった。なお，29 バイトのしきい値を用いた時の誤検知数は 374 であったが、漏洩情報を付加せずに同様の実験を行った際は 400 であった。また、780 バイトのしきい値を用いた時の誤検知数は 14 であったが、漏洩情報を付加しない場合は 17 であった。このことから、漏洩情報が付加されても誤検知数は大きくは変わらず、一定数の誤検知は避けられないと考えられる。

（６）まとめと今後の課題

本論文では漏洩検知の手段として、近似情報量を算出する手法とそれを用いた検知システムについて示した。近似情報量算出プログラムを実行した結果、編集距離を求める際にさかのぼる対象の HTTP リクエストを増やすことで古い情報を新しい情報としてカウントするのを防ぐことができることが確認できた。また、その応用として検知システムの提案と模擬実験を行った。検知システムは漏洩情報量が大きい時は誤検知率が 11.6%と誤検知数の少ないアラートを挙げることができたが、漏洩情報量が小さい時は誤検知率が 77.8%と誤検知数の多い結果となった。今後の課題は近似情報量算出プログラムと検知システムの改善である。近似情報量算出プログラムの改善手法としては、URL の動的生成法の実装等が挙げられる。また、文字を分割して編集距離を求めることによる、プログラム実行時間の短縮も検討する。検知システムは数値だけで判断するのではなく、送信される HTTP リクエストの挙動等の条件も合わせて総合的に判断できないかを検討する。

〈発　表　資　料〉

題　　名	掲載誌・学会名等	発表年月
履歴情報に基づく HTTP リクエストにおける情報漏洩量の数値化手法の検討	電気関係学会九州支部連合大会（第６４回連合大会）	2011 年 9 月
履歴情報に基づく HTTP リクエストにおける情報量の数値化手法の検討と検知システムの考案	コンピュータセキュリティシンポジウム 2011	2011 年 10 月
Towards Countermeasure Against Insider Threat on Network Security	Proceedings of the 3rd International Workshop on Managing Insider Security Threats (MIST 2011), pp.634-636, IEEE Computer Society	2011 年 12 月
Reviewing the Way to Quantifying Information Leaks on HTTP Requests, and Proposing the Detection System	Fifth Workshop among Asian Information Security Labs (WAIS’2012)	2012 年 1 月