情報処理学会論文誌 Vol.54 No.3 1071–1076 (Mar. 2013)

テクニカルノート

HTTPリクエストの情報量の異常値検出を用いた漏洩検知

千葉 一輝[1,2,a)] 堀 良彰[1,2,b)] 櫻井 幸一[1,2,c)]

受付日 2012年6月27日, 採録日 2012年10月10日

概要：情報漏洩問題に直面する機会が増加している．なかでも，ネットワークを介した情報漏洩に関しては対策が不十分である．漏洩の手法としてHTTPリクエストを利用したものがある．既存研究では直前のHTTPリクエストと送信リクエストの編集距離を測ることにより得られる近似情報量を提案している．本手法では新しく送信される情報のみが近似情報量として数値化されるため，異常検知が容易になる．本稿では，直前だけではなくさらに前のHTTPリクエストと比較をして編集距離を求めることで，近似情報量の精度を上げる手法を提案・評価した．本手法を用いて構成した漏洩検知システムを使った実験では，漏洩情報量が1,000バイトのときには誤検知率は11.6%であった．

キーワード：情報漏洩，HTTPリクエスト，異常検知，編集距離

Information Leakage Detection System Based on the Edit Distance of HTTP Requests

Kazuki Chiba[1,2,a)] Yoshiaki Hori[1,2,b)] Koichi Sakurai[1,2,c)]

Received: June 27, 2012, Accepted: October 10, 2012

Abstract: We often face the information leakage incident. However countermeasures are inadequate about information leakage on the Internet. Attackers use HTTP requests to theft information. Researches describe ways of calculating the approximate entropy to detect abnormal behavior. The approximate entropy is a value calculated by counting the edit distance of HTTP requests. We proposed the way to refer the historical HTTP requests to calculate the approximate entropy more accurately. Moreover, we designed the system raising alerts if information leakage comes into being. The system raises alerts whose false positive rate was 11.6% when the amount of leaked information was 1,000 bytes.

Keywords: information leakage, HTTP request, anomaly detection, edit distance

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，なか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としてスパイウェアが存在する．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履歴等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[1], [2]．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等が，個人情報を送信することによって漏洩が発生する．Pearce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

[1] 九州大学大学院システム情報科学府
Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka 819–0395, Japan
[2] 財団法人九州先端科学技術研究所
Institute of Systems, Information Technologies and nanotechnologies, Fukuoka 814–0001, Japan
a) chiba@itslab.inf.kyushu-u.ac.jp
b) hori@itslab.inf.kyushu-u.ac.jp
c) sakurai@csce.kyushu-u.ac.jp

おり，その中の 56%が本来は必要がないと思われるパーミッションを得ることが分かっている [3]．パーミッションを得たアプリは位置情報等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 [4]．近年では多くの Web アプリケーションが HTTP 通信を行っており，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 1,000 バイトのときにあ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あげられることが確認できた．

2. 関連研究

Borders 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 [5]．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 1 文字ずつ変換するのに必要な手順の回数である [6]．今回はこの手法により，すでに送られているデータはカウントされないことになり無視できる．com と co.jp の編集距離の算出方法を**図 1** に示す．この例では置換 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 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 [7]．編集距離を通常どお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．たとえば 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 $O(n^2)$ であるがゆえに高速となる．

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という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．

図 1 編集距離の求め方
Fig. 1 How to calculate the edit distance.

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，それを古い情報と見なすことが不可能である．

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 [8]．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 [9]．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

3.1 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

図 2 HTTP リクエストの例 1
Fig. 2 Example of the HTTP request (1).

```
GET /test.jpg HTTP/1.1
Host: example.com
```

図 3 HTTP リクエストの例 2
Fig. 3 Example of the HTTP request (2).

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．

3.2 履歴情報の参照

実験対象とする HTTP リクエストは無作為に選んだ同一の始点 IP アドレスから，PC で行う Web ブラウジングをする際に送信されるものである．観測開始から $k-2$，$k-1$，k 番目の HTTP リクエストの例を**図 2** に示す．まず直前の HTTP リクエスト（$k-1$ 番目）との各フィールドごとの編集距離を求める．k 番目の HTTP リクエストの「POST」，「I'm_busy.」という部分の編集距離はそれぞれ 3，9 となる．それ以外のフィールドは一致しているので編集距離は 0 となる．次に k 番目の HTTP リクエストと 2 つ前の HTTP リクエスト（$k-2$ 番目）とを比較する．「POST」，「I'm_busy.」という部分の編集距離はそれぞれ 0，4 となる．最後にそれぞれのフィールドで最も編集距離の小さかったものを合計する．「POST」の部分の編集距離はそれぞれ 3，0 であったので 0 とする．「I'm_busy.」の部分の編集距離はそれぞれ 9，4 であったので 4 とする．k 番目の HTTP リクエストの最終的な近似情報量は $0+4=4$（バイト）となる．以上が具体的な近似情報量算出方法である．なお，この例では比較対象は $k-2$ 番目，$k-1$ 番目の HTTP リクエストなので履歴参照数は 2 となる．履歴参照数が 3 以上のときも同様に比較対象を増やす．

3.3 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．HTTP リクエストの例を**図 3** に示す．

```
GET / HTTP/1.1
Host: example.com
Referer: http://example.com/test.jpg
```

図 4 HTTP リクエストの例 3
Fig. 4 Example of the HTTP request (3).

Host フィールドの example.com はドメイン名，/test.jpg はリクエスト URI である．したがってこの HTTP リクエストは http://example.com/test.jpg という URL から生成されたものである．この HTTP リクエストの直後に生成される HTTP リクエストを**図 4** に示す．Referer フィールドの URL は直前の HTTP リクエストの生成元の URL と一致しているので，新しい情報ではないと判断できる．したがって Referer フィールドの数値は 0 とする．一致しなかった場合は編集距離を求める．

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 1 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 1.60 GHz
- メモリ：512 MB

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

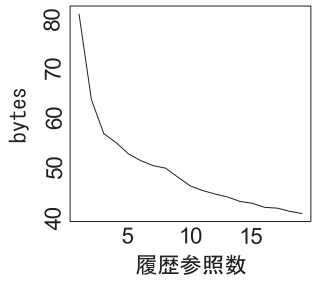

図 5 履歴参照数と近似情報量

Fig. 5 The number of referring to history and the approximate entropy.

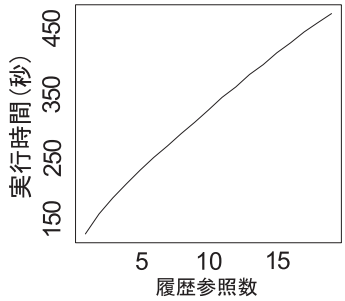

図 6 履歴参照数と実行時間

Fig. 6 The number of referring to history and the runtime.

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**図 5** に示す．x 軸は履歴参照数，y 軸は算出された平均近似情報量を表す．この結果より，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分かる．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したことで，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**図 6** に示す．これは図 5 の実験の際の実行時間を表すグラフである．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 11,259 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 リクエストが何個存在しているかを表すグラフを**図 7** に示す．このグラフよりリクエストサイズが 0 バイトから 3,000 バイト付近までの範囲に散らばっていることが分かる．このときの平均サイズは 940 バイトであった．

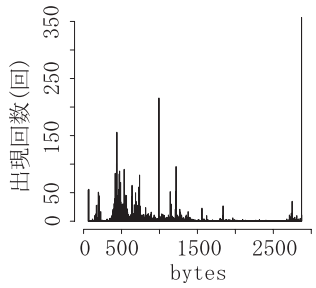

図 7 HTTP リクエストの文字数の分布

Fig. 7 Distribution of the number of the HTTP requests' character.

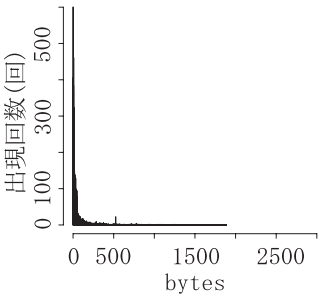

図 8 近似情報量の分布

Fig. 8 Distribution of the approximate entropy.

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**図 8** に示す．図 7 と比べ，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分かる．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．たとえば NAT 等が利用され，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 アドレスごと等，アプリケーションを識別する手法を適用したうえ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

表 1 誤検知率
Table 1 False positive rate.

履歴参照数	0	1	10
誤検知率（30 バイト）	90.0%	84.8%	77.8%
誤検知率（1,000 バイト）	86.7%	55.6%	11.6%

あげる．たと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．

これらをふまえて検知システムを作成する．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あげる．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．

(1) 無作為に選んだメンバ 1 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 1,069 個を収集する．
(2) 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，1,000 バイト）を付加する．
(3) 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 バイトのときは 29 バイト，漏洩情報量が 1,000 バイトのときは 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**表 1** に示す．

表 1 より，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 1,000 バイトと大きいときは誤検知率の比較的小さなアラートがあ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 バイトと小さいときは誤検知率が大きいという結果となった．なお，29 バイトのしきい値を用いたときの誤検知数は 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ときの誤検知数は 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

本稿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ときは誤検知率が 11.6%と誤検知数の少ないアラートをあげることができたが，漏洩情報量が小さいときは誤検知率が 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あげられる．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あわせて総合的に判断できないかを検討する．

謝辞 本研究の一部は，日本学術振興会「科学研究費補助金（B）（23300027）」による支援を受けている．また，本研究の一部は，総務省「国際連携によるサイバー攻撃の予知技術の研究開発（PRACTICE: Proactive Response Against Cyber-Attacks Through International Collaborative Exchange）」による支援を受けている．

参考文献

[1] 与那原亨，大谷尚通，馬場達也，稲田 勉：トラフィック解析によるスパイウェア検知システムの提案（セッション 6-B：不正検知），情報処理学会研究報告，CSEC［コンピュータセキュリティ］，Vol.2006, No.26, pp.197–202 (2006).
[2] Alme, C. and Pekrul, M.: SYSTEMS AND METHODS FOR MALWARE DETECTION, US Patent App. 13/021, 585, p.1 (2011).
[3] Pearce, P., Felt, A.P., Nunez, G. and Wagner, D.: AdDroid Privilege Separation for Applications and Advertisers in Android, *ASIACCS'12* (2012).
[4] 与那原亨，大谷尚通，馬場達也，稲田 勉：トラフィック解析によるスパイウェア検知の一考察，電子情報通信学会技術研究報告，ISEC，情報セキュリティ，Vol.105, No.193, pp.23–29 (2005).
[5] Borders, K. and Prakash, A.: Quantifying information leaks in outbound web traffic, *30th IEEE Symposium on Security and Privacy 2009*, pp.129–140, IEEE (2009).
[6] Ristad, E. and Yianilos, P.: Learning string-edit distance, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.20, No.5, pp.522–532 (1998).
[7] Borders, K.: *Protecting confidential information from malicious software*, p.108, The University of Michigan (2009).
[8] Borders, K.: Products – Web Tap Personal, available from ⟨http://webtapsecurity.com/products_personal.html⟩ (accessed 2012-03-15).
[9] Borders, K. and Prakash, A.: Web tap: Detecting covert web traffic, *Proc. 11th ACM Conference on Computer and Communications Security*, pp.110–120, ACM (2004).

千葉 一輝 （学生会員）

平成元年生．平成 23 年九州大学工学部電気情報工学科卒業．同年同大学大学院システム情報科学府情報学専攻修士課程入学．

堀 良彰 （正会員）

昭和 44 年生．平成 4 年九州工業大学情報工学部電子情報工学科卒業．平成 6 年同大学大学院情報工学研究科情報システム専攻修士課程修了．平成 6 年九州芸術工科大学助手．博士（情報工学）．平成 16 年より九州大学大学院システム情報科学研究院助教授（現，准教授）．平成 17 年から 18 年にかけてカリフォルニア大学アーバイン校計算機科学部訪問研究員．情報ネットワーク，ネットワークセキュリティ，コンピュータシステムセキュリティ等の研究に従事．電子情報通信学会，IEEE，ACM 各会員．

櫻井 幸一 （正会員）

昭和 38 年生．昭和 61 年九州大学理学部数学科卒業．昭和 63 年同大学大学院工学研究科応用物理専攻修士課程修了．昭和 63 年三菱電機株式会社入社後，情報電子研究所（現，情報総合研究所）にて，暗号と情報セキュリティの研究開発に従事．博士（工学）．平成 6 年より九州大学工学部情報工学科助教授．平成 9 年から 10 年にかけて米国コロンビア大学計算機科学科訪問研究員．現在，九州大学大学院システム情報科学研究院教授．平成 16 年より財団法人九州システム情報技術研究所第 2 研究室長（現，財団法人九州先端科学技術研究所情報セキュリティ研究室長）．電子情報通信学会，日本数学会，ACM，IEEE 各会員．