HTTP リクエストにおける情報漏洩量の数値化手法の検討と検知システムの提案

千葉 一輝 †　　堀良彰 †　　櫻井幸一 †

† 九州大学大学院　システム情報科学府/財団法人九州先端科学技術研究所
819-0395 福岡市西区元岡 744/814-0001 福岡市早良区百道浜 2-1-22

chiba@itslab.csce.kyushu-u.ac.jp
{hori,sakurai}@inf.kyushu-u.ac.jp

あらまし 情報の漏洩問題に直面する機会が増えてきている．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．攻撃者が HTTP リクエストに欲しい情報を埋め込むことで情報漏洩が発生する．そこで HTTP リクエストにおける情報漏洩検知システムについて考える．既存研究では編集距離の考えを用いて，繰り返される冗長な部分を無視して文字数を測ることにより得られる近似情報量を調べている．本研究では，既存研究の考えを発展させ，さらに精度のよい近似情報量を求めるための手法を提案・評価した．さらに近似情報量を測定するシステムを用いて，情報漏洩の恐れがある際にアラートを挙げるシステムの提案を行った．

Reviewing the way to quantifying information leaks on HTTP requests, and proposing the detection system

Kazuki Chiba†　　Yoshiaki Hori†　　Kouichi Sakurai†

†Graduate School of Information Science and Electrical Engineering, Kyushu University/
Institute of Systems, Information Technologies and nanotechnologies
744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan/
2-1-22 Momochihama, Sawara-ku, Fukuoka 814-0001
chiba@itslab.csce.kyushu-u.ac.jp
{hori,sakurai}@inf.kyushu-u.ac.jp

Abstract We often face the problem of information leaks, but countermeasures are inadequate about information leaks on the Internet. Embedding information in HTTP requests by attackers leads to information leaks. Therefore, we consider detection system for information leaks on HTTP requests. Existing research describes ways of measuring the approximate entropy. The approximate entropy is a value calculated by counting the number of the new information characters. We designed and implemented the system based on the existing research that measures sensitive approximate entropy. Moreover, we designed the system that raises alerts if information leaks come into being.

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．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十分ではない．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というものが増えている [1]．これらスパイウェアには HTTP リクエストが平文かつ書き換えが容易というという性質を利用してそれを書き換え，HTTP リクエストをスパイウェアサーバに送信することで漏洩を引き起こすものがある．PC にインストールされたスパイウェアは，HTTP を利用してスパイウェア用サーバと双方向かつリアルタイムな通信を行う [3]．Gator というスパイウェアは過去に閲覧した URL を GET リクエストを使用し，リクエスト URI の部分に閲覧した Web ページの URL 等を埋め込みスパイウェアサーバに送信する．また，検索やログイン等で使う入力フォームを使って送信した文字を，POST リクエストの body 部分に含めてスパイウェアサーバに送信する [2]．本研究では以上のような漏洩を含む，HTTP リクエストを用いた情報漏洩を検知するシステムで必要となる漏洩情報量の数値化手法を検討する．

2 関連研究

Kevin Borders，Atul 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試みている [4]．数値化の１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．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．Kevin 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 [5]．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．

com → (置換)co. → (挿入)co.j → (挿入)co.jp

図 1: 編集距離の求め方

なお 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からである．Host フィールドは接続するサーバを示す．Host フィールドが直前に見ていたページの HTML にある，リンク 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，大幅な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はそれを無視することができないと考えられる．

また，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 [6]．

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，掲示板等に自発的に書き込んだ際の漏洩は対象としていないと思われる [7]．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　提案手法

HTTP を利用した漏洩を検知するために関連研究の数値化手法を使用する．その利点を図 2 に示す．この図は送信される HTTP リクエストのサイズと，数値化したサイズを表す．数値化されたサイズを本論文では近似情報量と呼ぶ．HTTP リクエストサイズをただ測定するだけでは，漏洩情報が含まれていても発見するのが困難である．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，この手法ではサイズが際立つことになり発見が容易になる．

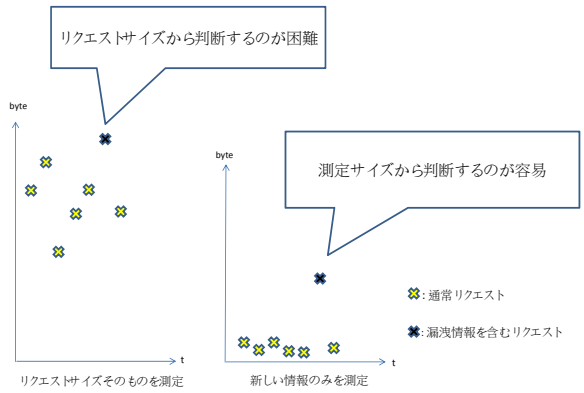

図 2: 近似情報量を求める利点

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，n 個前までの HTTP リクエストとの編集距離を求めることで，古い情報をさらに無視することを試みる．具体的な方法を図 3 に示す．

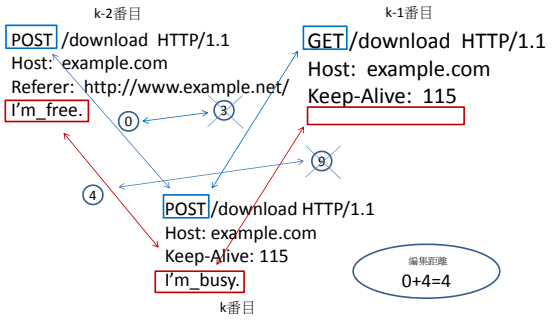

図 3: 具体的な測定方法

図 3 は観測開始から k 番目の HTTP リクエストの近似情報量算出を試みている例である．この例では比較対象は 2 つ前までの HTTP リクエスト ($k-2$ 番目)，つまり $n=2$ となる．まず直前の HTTP リクエスト ($k-1$ 番目) との各フィールド毎の編集距離を求める．k 番目の HTTP リクエストの POST，I'm_busy. という部分はそれぞれ編集距離は 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) と比較をする．POST，I'm_busy. という部分はそれぞれ編集距離は 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 の部分では編集距離がそれぞれ 3，0 であったので 0 とする．I'm_busy. の部分では編集距離がそれぞれ 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 $0+4=4$ となる．以上が具体的な近似情報量算出方法である．今回は $n=2$ の場合を例にしたが，$n>2$ のときも同様に比較対象を増やす．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，フィールドは書き換えられることで漏洩情報が含まれていることがないと判断できる．その場合は数値は 0 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．具体的な手法を示す．HTTP リクエストの例を図 4 に示す．

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

図 4: HTTP リクエストの例

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 というリンク 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 5 に示す．Referer フィールドの URL は直前の HTTP

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

図 5: HTTP リクエストの例 2

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 フィールドは以前に送った Cookie フィールドを保存しておき，それと一致するものがあれば数値を 0 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 の部分は HTTP レスポンスに含まれる HTML の内容から URL 部分を正規表現にて抽出し，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

比較対象の個数 n が算出される数値とどのような関係があるかを調べるために実験を行う．実験対象は 4 つの IP アドレスから送信される HTTP リクエスト 500 個である．HTTP リクエストは 2011 年 4 月の同じ研究室の人のものを収集した．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．今回の実験では n の値は 1 から 19 とした．n の値と近似情報量との関係を図 6 から図 9 に示す．この結果より，n の値を増やすことで数値が低くなっていることがわかる．これは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，さらに古い情報を無視することができたと考えられる．

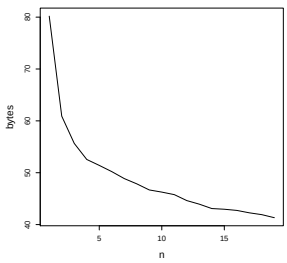

図 6: n と近似情報量の関係 1

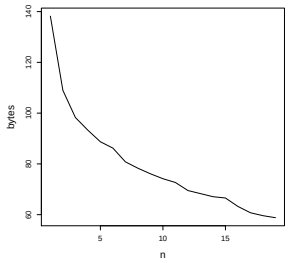

図 7: n と近似情報量の関係 2

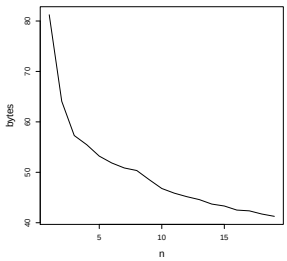

図 8: n と近似情報量の関係 3

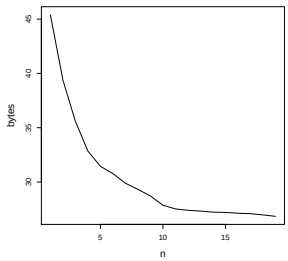

図 9: n と近似情報量の関係 4

また，n と本プログラムの実行時間の関係を図 10 に示す．これは図 6 の実験の際の実行時間を表すグラフである．n の値が増加するのに比例して実行時間も増加している．図 7-図 9 の

実験の実行時間もほぼ同じ結果となった．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不可能であるといえる．

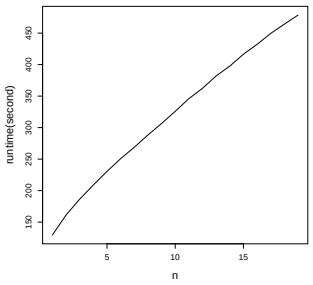

図 10: n と実行時間の関係

次に，本プログラムを実行す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．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 11 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 3000 バイト付近までの範囲に散らばっていることがわかる．

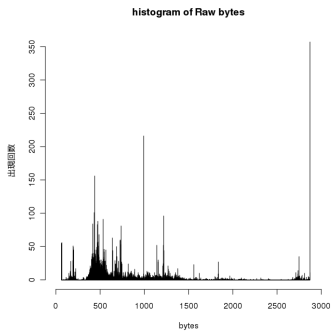

図 11: HTTP リクエストの文字数の分布

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 n の値は 10 とする．各近似情報量の存在する数の分布を図 12 に示す．図 11 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わかる．編集距離を用いた数値化手法をとることで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．

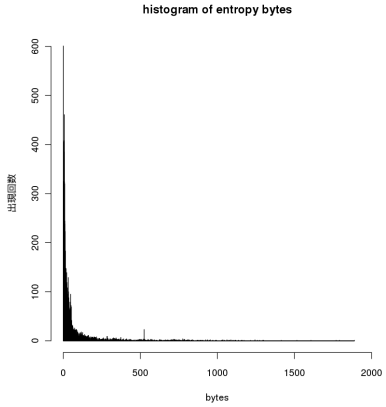

図 12: 近似情報量の分布

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．

1. 正常 HTTP リクエストを 1500 個用意する．
2. その 1 割の 150 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．

3. システムでHTTPリクエストを解析し，正検知数，誤検知数を調べる

しきい値は理想的なもの (未検知数が0かつ誤検知数が最も少なくなる値) を設定する．30バイトという短い文字列を付加した際は正検知数は150個となるが，誤検知数が553個となった．アラートを挙げた際にそれが漏洩情報に対してのアラートとなる確率は21.3%である．それに対して，1000バイトという長い文字列を付加した際は正検知数は150個，誤検知数は11個となった．漏洩情報に対してのアラートとなる確率は93.1%である．以上のことから漏洩情報量が大きい時は精度の高いアラートを挙げることができているといえる．それに対して情報漏洩量が小さい時は誤検知数が多くシステムとしては不十分であるといえる．

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必要な，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することで，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精度の高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数が多く十分な結果が得られなかった．今後の課題は近似情報量算出プログラムの改善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，関連研究で調べていた動的に生成されるURLについての実装が挙げられる．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考える．検知システムは数値だけで判断するのではなく，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考える．

7 謝辞

本研究の一部は国際連携によるサイバー攻撃の予知技術の研究開発 (総務省) の支援を受けたものである．

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”，情報処理学会研究報告，*2006-DPS-126 (34) 2006-CSEC-32 (34), pp.197-202*, 2006/3/17

[2] 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報告，*ISEC2005-12,SITE2005-10(2005-07), pp.23-29*, 2005/07/21

[3] 大谷 尚通，与那原 亨，馬場 達也，稲田 勉，HTTP利用型スパイウェアの検知および遮断方式の検討情報処理学会 コンピュータセキュリティ *(CSEC)* 研究会，*2005-CSEC-31 (3), pp.13-18*, 2005/12/9

[4] Kevin Borders, Atul Prakash, “Quantifying Information Leaks in Outbound Web Traffic”, *proceeding of the 30th IEEE Symposium on Security and Privacy, pp.279-287*, November 2007

[5] Kevin Borders, “ Protecting Confidential Information from Malicious Software”, *Ph. D. Thesis, pp.101-102*, May 2009.

[6] “Products - Web Tap Personal”, http://www.webtapsecurity.com/products_personal.html

[7] Kevin Borders and Atul Prakash, “Web Tap: Detecting Covert Web Traffic.”, *Proceedings of the 11th ACM Conference on Computer and Communications Security (CCS), pp.3*, Oct. 2004.