Computer Security Symposium 2019
21 - 24 October 2019

リソース統計情報を用いたフィッシングサイト検知

田中 翔真[1,a)]　松中 隆志[1,b)]　山田 明[1,c)]　窪田 歩[1,d)]

概要：近年，フィッシングによる個人情報の詐取が増加しており対策が急務となっている．フィッシングサイトは，銀行サイトなど正規のサイトを模擬して構築されるため，見た目が類似しているという特徴がある．サイト間の見た目の類似度を測定するため，スクリーンショットを取得して比較する検知手法が提案されている．しかし，スクリーンショット取得は，サイトへのアクセスやコンテンツの取得など処理に時間がかかるという課題がある．本稿では，Web サイトを構成するコンテンツの種類・総数・サイズなどリソース統計情報のみを用いたフィッシングサイト検知手法を提案する．提案手法は，アクセスログのみを利用して実装できるため，従来手法と比較して処理量が少なく，Web プロキシサーバなど様々な監視地点に実装できる．提案方式の有効性を検証するために，ユーザ参加型 Web 媒介型攻撃対策の実証実験によって収集したログ及び既知のフィッシングサイトにアクセスして収集したログを用いて評価を行った．その結果，提案手法はスクリーンショットの比較による検知手法に近い性能が得られることが分かった．

キーワード：フィッシングサイト検出，Web アクセスログ分析

Phishing Sites Detection using Statistical Information of Resources

Shoma Tanaka[1,a)]　Takashi Matsunaka[1,b)]　Akira Yamada[1,c)]　Ayumu Kubota[1,d)]

Abstract: Stealing personal information by phishing is increasing, and it is necessary to take measures. Since phishing sites are constructed by simulating regular sites and they are similar in appearance. In order to measure the visual similarity between websites, a detection method compares screenshots has been proposed. However, it takes time to access sites and acquire contents to acquire screenshots. In this paper, we propose a phishing site detection method that uses resource statistics that compose a website. The proposed method can be implemented using only web access logs, so the amount of processing is smaller compared to the conventional method. In order to verify the effectiveness of our proposed method, we evaluate using logs collected by our demonstration experiment and logs collected by accessing known phishing sites. As a result, it was found that our proposed method can perform similar to the detection method compares screenshots.

Keywords: Phishing Site Detection, Web Access Logs Analysis

1. はじめに

フィッシングとは，個人情報を詐取しようとする試みのことであり，フィッシングサイトとは，上記の目的を達成するために構築された Web サイトのことである．近年のインターネットの急速な普及に伴い，フィッシングサイトによる個人情報詐取被害が増加してきている．フィッシングサイトは基本的には正規サイトを模倣し，正規サイトの信頼性を悪用して個人情報の詐取を試みる．フィッシングサイトの検知手法として，フィッシングサイトの特徴を用いた以下の方法が考えられてきた．

- ブラックリストに基づく検知
- URL の類似性に基づく検知
- スクリーンショットの類似性に基づく検知

各手法の詳細については 2 章で述べる．

[1] KDDI 総合研究所
KDDI Research, Inc.
[a)] sm-tanaka@kddi-research.jp
[b)] ta-matsunaka@kddi-research.jp
[c)] ai-yamada@kddi-research.jp
[d)] kubota@kddi-research.jp

本稿では，Web ページの構成におけるリソース統計情報に基づくフィッシングサイトの検知手法を提案する．ここでリソース統計情報とは，Web サイトを構成するコンテンツの種類・総数・サイズ等のことである．フィッシングサイト検出手法として，Web サイトのコンテンツ情報をベースとした検出手法が提案されており [1]，この手法では，Web ページの特徴を表す語句を基にフィッシングサイトの検出を試みている．そこで筆者らは，コンテンツ情報の代わりに Web ページを構成する際にアクセスするリソースの統計情報のみを用いてフィッシングサイトの検出を試みる．本手法はスクリーンショットの取得による手法と比較して解析の際の処理量が少ないという特徴があるため，これまでの課題であった視覚的特徴の類似するサイトの判定及び取得に時間がかかるという課題を解決できる．

本稿では，まず 2 章において，既存のフィッシングサイト手法について述べる．3 章において筆者らの提案するフィッシングサイト検知手法を示す．4 章において PhishTank による提案手法の評価を行い，5 章において筆者らの提案しているユーザ参加型 Web 媒介型攻撃対策の実証実験データによる提案手法の評価を行う．6 章において考察を述べ，7 章においてまとめ及び今後の課題について述べる．

2. 既存手法

フィッシングサイトの検知手法として，フィッシングサイトの特徴を用いた以下の方法が考えられてきた．

2.1 ブラックリストに基づく検知

フィッシングサイトであると判定された URL の情報をブラックリストとして保持し，アクセスした URL がブラックリストに登録されている場合にその URL へのアクセスを遮断する方式である．しかし，この方式の場合には，リストの更新が追い付かないという課題が存在する．攻撃者は構築したフィッシングサイトがブラックリストで検出されることを避けるために，新たなフィッシングサイトをドメインを変えて再構築し，以前のフィッシングサイトを閉鎖するというサイクルを短期間の間に繰り返し行う．したがって，一般的にフィッシングサイトの生存期間は極めて短いものとなっている．これにより，あるフィッシングサイトがブラックリストに登録され，その情報がユーザサイドに反映される頃には該当サイトが閉鎖されているということが起こりうる．

2.2 URL の類似性に基づく検知

同一正規サイトを模倣したフィッシングサイトに関して，URL に類似性が見られる場合がある．例えば，2018 年ごろから被害が急増した佐川急便を装う偽サイトは，佐川急便の正規サイトを模倣して構築されており，URL のドメインに以下の特徴が見られる [2].

- **sagawa-xxx.com**

ここで，xxx はランダムに構成された文字列である．また，ドメインの生存期間は典型的なフィッシングサイトと同様に短い．上記のようなフィッシングサイトを検知する方法として，正規表現を用いて既知のフィッシングサイトの URL との類似性を基に検知する方法が挙げられる [3]．しかしながら，同じ正規サイトを模倣して構築されたフィッシングサイトの URL の文字列が類似しているという想定をしているため，佐川偽サイトのような URL の文字列にパターンの見られるフィッシングサイトは検知可能であるが，URL の文字列が類似しないフィッシングサイトは検知できないという課題が存在する．

2.3 スクリーンショットの類似性に基づく検知

上記の課題を解決するために，スクリーンショットの類似性を基にフィッシングサイトを検知する手法が提案されている [4]．ブラウザが Web ページにアクセスした際のスクリーンショットを取得し，画像間の類似度を測定し，閾値以上の類似度の場合にフィッシングサイトとして検知する方式である．しかしながら，すべての Web サイトに対してスクリーンショットの取得，画像間の類似度の判定が必要となり，処理に時間がかかるという課題が存在する．

3. Web ページの構成におけるリソース統計情報の一致性に基づくフィッシングサイト検出手法

本章では，筆者らの提案するフィッシングサイト検出手法について述べる．

3.1 フィッシングサイト検知におけるシステム構成図

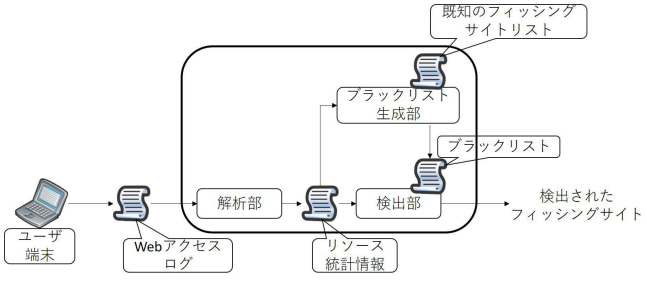


図 1 フィッシングサイト検知におけるシステム構成図

フィッシングサイト検知におけるシステム構成図を図 1 に示す．ユーザ端末から Web アクセスログを収集して解析部に送信する．Web アクセスログは，ユーザ端末のブラウザ上のプラグインにより生成される．解析部ではリソース統計情報を生成してブラックリスト生成部と検出部

に送信する．ブラックリスト生成部では解析部で生成したリソース統計情報の URL が既知のフィッシングサイトである場合に当該リソース統計情報をブラックリストとして記録する．検出部では解析部で生成したリソース統計情報がブラックリストに含まれる場合に当該 URL を未知のフィッシングサイトとして検知する．

3.2 解析に用いる情報について

表 1 収集情報

収集情報	説明
userid	ユーザを一意に識別する ID
tabid	ブラウザタブを一意に識別する ID
url	リソースの URL
type	リソースの種類
content-length	リソースの content-length
malicious_info	リソースの URL の悪性情報

解析には，**表 1** の情報で構成される Web アクセスログを用いる．また解析環境として，図 1 のようなシステム構成を想定している．

userid は，Chrome のプラグインにより一意に生成される．

tabid，url，type は，Chrome の webRequest API[5] により取得する．また，type の詳細を**表 2** に示す．

content-length は，HTTP ヘッダより取得する．

malicious_info は，フィッシングサイトであるか否かの情報である．リソースの URL のフィッシングサイト判定には，Google Safe Browsing[6]（以下 GSB）を用いる．GSB のラベルが SOCIAL_ENGINEERING である場合にフィッシングサイトと判定する．GSB のラベルが SOCIAL_ENGINEERING でない場合はフィッシングサイトではないと判定する．

3.3 Web ページの構成におけるリソースの統計情報

ある Web ページ（https://www.example.com/）を構成する際の Web アクセスログは**表 3** のようになる．ある Web ページにアクセスした場合，まずタブにロードされるトップレベルのドキュメントである main_frame の URL にアクセスし，その後，Web ページを構成する各リソース（stylesheet，script，image，image，image）の URL へアクセスする．ここで，ある main_frame の URL へのアクセスに関して，次の main_frame の URL にアクセスするまでにアクセスしたリソースの URL の統計情報を取得する．具体的には，下記の 3 種類の情報を取得する．

- main_frame を除くリソースごとの総アクセス数 A
- image リソースの content-length の総和 c
- main_frame を除くリソースごとのユニークドメイン数 D 及び全リソースのユニークドメイン数

これらの統計情報を組み合わせた下記の 3 種類のリソース統計情報を定義する．

- リソース統計情報 1：[A，c]
- リソース統計情報 2：[A]
- リソース統計情報 3：[A, D]

尚，表 3 の Web ページを構成する際のリソースの統計情報に関して，リソースの URL の統計情報は下記の通りである．

- A=[0, 1, 1, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0]
- c=15000
- D=[0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]

但し，A に関して，表 2 で記載した順に数えることとした．また，D に関して，表 2 で記載した順に数え，全リソースのユニークドメイン数を最後に記載することとした．

表 3 の例におけるリソース統計情報 1，2，3 を**表 4** に示す．

3.4 リソースの統計情報に基づくフィッシングサイト検出手法

典型的なフィッシングサイト構築において，攻撃者は正規サイトを模倣する．また，ブラックリストによる検知を避けるため，フィッシングサイトの生存期間は極めて短い．上記の特徴から，攻撃者がフィッシングサイト群の構成を変えないと仮定すると，ある正規サイトを模倣したフィッシングサイト群のリソース統計情報は一致していると考えられる．本手法では，同じ正規サイトを模倣したフィッシングサイト群はリソース統計情報が一致するという仮定に基づきフィッシングサイトの検知を試みる．

3.4.1 フィッシングサイト検知手順

(1) main_frame から main_frame の間の Web アクセスログを取得する．

(2) (1) で取得した情報を userid ごと，tabid ごとにソートする．

(3) 各 main_frame の URL に対して，当該サイトの構成におけるリソース統計情報 1，リソース統計情報 2，リソース統計情報 3 を作成する．

(4) (3) で観測された各 main_frame の URL が既知のフィッシングサイトである場合，当該サイトの構成におけるリソース統計情報をブラックリストとして記録する．

(5) (3) で観測された各 main_frame の URL のリソース統計情報がブラックリストにマッチし，既知のフィッシングサイトではない場合に当該 URL を未知のフィッシングサイトとして検知する．

ここで，既知のフィッシングサイトの URL の取得先として，Web アクセスログの malicious_info の情報を用いることを想定している．また，フィッシングサイトの URL を提供するサービスを用いる運用も考えられる．尚，誤検

表 2　type 詳細

type	説明
main_frame	タブにロードされるトップレベルのドキュメント
sub_frame	<iframe>要素または<frame>要素にロードされたドキュメント
stylesheet	ドキュメントの表現を記述するために読み込まれた CSS スタイルシート
script	<script>要素によって実行されるまたはワーカーで実行されるようにロードされるコード
image	そのタイプをサポートするブラウザのイメージセットを除いてイメージはイメージとしてレンダリングされるように読み込まれたリソース
font	@ font-case CSS ルールのために読み込まれた Web フォント
object	<object>要素または<embed>要素によってロードされるリソース
http_request	HTTP_Request によって送信されたリクエスト
xmlhttprequest	XMLHTTPRequest オブジェクトまたは FetchAPI によって送信されたリクエスト
ping	ハイパーリンクが続いたときにハイパーリンクの ping 属性で指定された URL に送信されたリクエスト
csp_report	ポリシーに違反する試みが検出されたときに Content-Security-Policy ヘッダーで指定された report-uri に送信された要求
media	<video>要素または<audio>要素によって読み込まれたリソース
websocket	WebSocketAPI を介してサーバへの接続を開始するリクエスト
other	その他の要素

表 3　ある Web ページを構成する際の URL 群へのアクセスに対応する Web アクセスログ

userid	tabid	url	type	content-length	malicious_info
user001	0	https://www.example.com/	main_frame	10000	None
user001	0	https://www.example.com/layout.css	stylesheet	20000	None
user001	0	https://www.example.com/top.js	script	30000	None
user001	0	https://www.example.com/example1.png	image	4000	None
user001	0	https://www.example.com/example2.png	image	5000	None
user001	0	https://www.example.com/example3.png	image	6000	None

知を減らすため，作成される各リソース統計情報に対して下記のフィルタを適用する．

フィルタ 1
リソース image の総和が閾値 th_1 以下の場合は検知対象外とする．(Not Found や Connection timed out などの少数の画像を用いた Web ページを除外するため．)

フィルタ 2
ドメインの出現回数が閾値 th_2 以上の場合は検知対象外とする．(Google などのアクセス数の多い Web ページを除外するため．)

フィルタ 3
image の content-length の総和が閾値 th_3 未満の場合は検知対象外とする．(Not Found や Connection timed out などのサイズの小さい画像を用いた Web ページを除外するため．)

但し，フィルタ 2 に関して，1 か月間のデータにおける出現回数を基にフィルタを実施するため，予め 1 か月分のデータを解析する必要がある．今回は閾値として，$th_1 = 2$，$th_2 = 10$，$th_3 = 1000$ を用いることとした．

4. PhishTank による事前評価

3.4.1 節で述べた手法に関して，リソース統計情報の一致する URL の視覚的特徴が一致していることを確認するために，PhishTank に投稿されたフィッシングサイト URL による評価を実施した．PhishTank[7] とは，フィッシングサイトに関する情報を共有するための無料のコミュニティサイトである．現在まで累計 15 万人のユーザにより 600 万件を超えるフィッシングサイトに関する情報提供がなされている．共有されたフィッシングサイトはユーザ同士によって検証され，ユーザの投票によって最終的な悪性判定がなされる．PhishTank に投稿されたフィッシングサイトに関する情報は誰でも利用可能であり，投稿された URL とその投票結果及び現在稼働中かの情報を取得可能である．フィッシングサイトの特徴に基づく仮定より，リソース統計情報が一致するようなフィッシングサイト群が多数抽出されると考えられる．

4.1 評価概要

PhishTank データのうち，2019 年 7 月 23 日時点で稼働中と判定されており，PhishTank ユーザの投票によって悪性と判定された 2019 年 6 月の 2,464URL を評価に用いた．提案手法により構造が一致しているフィッシングサイトとして抽出されたフィッシングサイト群に関してスクリーンショットを取得し，実際のサイトの視覚的類似性を併せて調査する．

表 4　ある Web ページを構成する際のリソース統計情報

	リソース統計情報
リソース統計情報 1	[[0, 1, 1, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0], 15000]
リソース統計情報 2	[[0, 1, 1, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
リソース統計情報 3	[[0, 1, 1, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]]

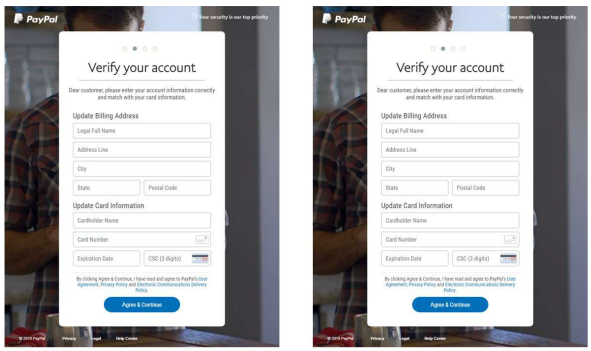
4.2　評価手順

(1) PhishTank データのフィッシングサイトの URL へアクセスし，ブラウザセンサによる Web アクセスログの取得を行う．
(2) 取得した Web アクセスログを用いて，各フィッシングサイトのリソース統計情報 1，リソース統計情報 2，リソース統計情報 3 を作成する．
(3) 各フィッシングサイトのリソース統計情報の比較を行い，統計情報の一致しているフィッシングサイトを Web ページ構造の一致したフィッシングサイトとして抽出する．

4.3　評価結果

表 5　提案手法による抽出 URL 数

	抽出 URL 数
リソース統計情報 1	650
リソース統計情報 2	979
リソース統計情報 3	932



https://multipromerchants.com/…　http://www.kisumuploy.ac.ke/…

図 2　抽出されたフィッシングサイト一例

Web ページ構造の一致したフィッシングサイトの抽出結果を**表 5** に示す．また抽出例として，リソース統計情報 1 により抽出されたフィッシングサイトの 1 例を**図 2** に示す．但し，URL は FQDN までの記載とした．図 2 のように URL の類似しない 2 つのフィッシングサイトにおいてスクリーンショットの視覚的特徴が一致していることがわかる．その他の事例について調査した結果，リソース統計情報 1 を用いて抽出された URL 群に関しては概ね視覚的特徴が一致していることが分かった．しかしながら，リソース統計情報 2，リソース統計情報 3 を用いて抽出された URL 群に関してはスクリーンショットの視覚的特徴が異なる事例が多数確認された．したがって，リソース統計情報をそのまま視覚的特徴の一致するフィッシングサイト検知に適用するような場合には，リソース統計情報 1 を用いる手法が有効であると考えられる．

4.4　スクリーンショットの一致による評価手法との比較

提案手法により視覚的特徴の一致しているサイト群が網羅的に抽出できているかを評価するために，スクリーンショットの一致性による抽出を実施し，リソース統計情報 1 を用いた提案手法の抽出結果との差異について評価した．尚，スクリーンショットの一致性の調査には Average Hash を用いた．

- Average Hash 生成手順

(1) 画像サイズを 64 × 64 に圧縮
(2) 画像をグレースケールに変換
(3) 画素値の平均値を算出
(4) 各画素の値が平均より上なら 1，下なら 0 に変換

4.4.1　評価手順

あるフィッシングサイトのスクリーンショットの組に対して Average Hash を算出し，Average Hash が一致した場合に，当該 URL の組を視覚的特徴の一致するフィッシングサイトとして抽出する．

4.4.2　評価結果

表 6　リソース統計情報 1 及び Average Hash の一致性による抽出 URL 数

	リソース統計情報 1	Average Hash
抽出 URL 数	650	715

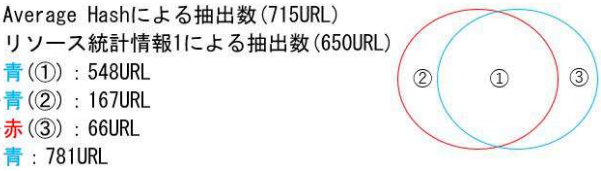


図 3　リソース統計情報 1 及び Average Hash による抽出 URL の関係

結果を**表 6** に示す．また，**図 3** に提案手法及び Average Hash により抽出された URL の関係を示す．結果より，視覚的特徴が一致している Web ページとして抽出すべき URL に対して，実際に抽出できた割合は，①/(①+②)=76.6% となる．

②の部分の関しては，スクリーンショットが完全に一致しているが，リソース統計情報が一致しなかった事例である．

③の部分に関しては，content-length が一致しているため，本来スクリーンショットの一致性により抽出されるべき事例である．詳細を調査した結果，スクリーンショットの取得の不具合が原因でスクロールバーが表示される場合とされない場合があることが判明した．そのためスクリーンショットの一致による抽出では抽出されなかったと考えられる．

以上より，提案手法はスクリーンショットの比較による手法に近い性能が得られることが確認されたため，本手法を実証実験データに適用し，フィッシングサイトの検知を試みる．

5. WarpDrive 実証実験データセットによる評価

本章では，実証実験データセットを用いて提案手法の評価を行う．既知のフィッシングサイト情報には，PhishTank と GSB の 2 種類を用いることとした．また併せて，フィッシングサイト検知により有効なリソース統計情報に関する検討を行った．

5.1 WarpDrive 実証実験データセット

5.1.1 WarpDrive

筆者らは，Web 媒介型攻撃対策技術の実用化に向けた研究開発（WarpDrive:Web-based Attack Response with Practical and Deployable Research InitiatiVE）[8] を実施している．WarpDrive では，ユーザが使用する Web ブラウザにブラウザセンサと呼ばれるソフトウェアをインストールすることによって悪性サイトの観測網を構築する．

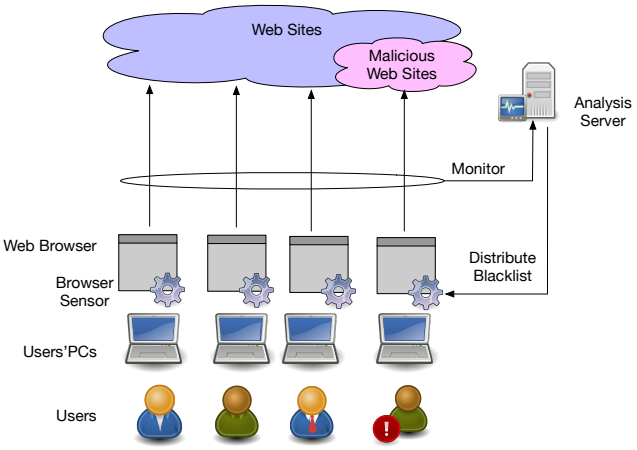


図 4　ユーザ参加型 Web 媒介型攻撃対策のシステム構成

図 4 にシステム構成図を示す．ユーザが遭遇する攻撃サイトの情報を観測することで，これまで全体の把握が困難であったユーザの操作が関わる Web 上の脅威について観測できることが期待できる．ブラウザセンサでは，観測網から得られた情報を元にした対策を実装しており，ユーザが悪性サイトの被害に遭遇することを防止する．このフレームワークにより，参加するユーザが増加するにつれて網羅的に悪性サイトを観測することが可能となる．

5.1.2 実証実験データセット

5.1.1 節で述べた WarpDrive の有効性を検証するために，2018 年 6 月 1 日から実証実験を実施している．実証実験では，実験を説明する Web サイト [8] を構築して，プレスリリースによって参加ユーザを募っている．2019 年 1 月 17 日において，登録 ID 数が 7,092 に達しており，1 日あたりのアクティブユーザは約 1,000 名である [9]．

実証実験においてユーザから収集するデータには，ブラウザデータ，web サイトアクセスデータ，コンテンツ，PC 環境情報等がある．詳細は実証実験参加規約 [10] に記載している．

今回の評価に用いたデータは，2019 年 5 月及び 2019 年 6 月の実証実験データである．

上記の URL に対して既知のフィッシングサイト URL との突合を実施した．2019 年 5 月の実証実験データと GSB を突合した結果，23URL がフィッシングサイトとして検知された．2019 年 6 月の実証実験データと 4.1 で用いた PhishTank データの URL を突合した結果，フィッシングサイトとして検知された URL は存在しなかった．

5.2 提案手法による検出

5.2.1 既知のフィッシングサイト情報に PhishTank を用いる場合

既知のフィッシングサイト情報に 4.1 の PhishTank データを用いて作成したリソース統計情報による未知のフィッシングサイトの検知を実施した．使用した実証実験データは 2019 年 6 月のデータである．尚，上記の PhishTank データと実証実験データの突合結果は 0 件であった．

表 7　リソース統計情報 1，2，3 による検知未知フィッシングサイト数 (2019 年 6 月)

	検知した未知フィッシングサイト数	リソース統計情報数
リソース統計情報 1	5	57,687
リソース統計情報 2	3,022	59,131
リソース統計情報 3	1,569	59,131

リソース統計情報 1 とリソース統計情報 2，3 に関して，適用されるフィルタが異なるため，フィルタ適用後の main_frame 数が異なることに注意が必要である．（リソース統計情報 1 は image の content-length の情報を用いているため，フィルタ 3 が適用されるが，リソース統計情報 2，

3 は image の content-length の情報を用いていないため，フィルタ 3 が適用されない．）

リソース統計情報 1，2，3 により検知した未知フィッシングサイト数を**表 7** に示す．リソース統計情報 2，リソース統計情報 3 に関してリソース統計情報 1 と比較した場合，検知数が多いことがわかる．しかしながら，スクリーンショットの比較結果より，誤検知がほとんどであることが分かった．したがって，リソース統計情報 2，リソース統計情報 3 をそのままブラックリストとしてフィッシングサイト検知に用いるのではなく，何らかの観点からフィッシングサイト検知に有効なリソース統計情報を絞り込む必要がある．5.2.2 節で上記に関する検討結果について述べる．

リソース統計情報 1 により検知された URL は 5URL であった．それらの URL のうち，1 件は正規銀行サイトを模倣したフィッシングサイトであった．残りの 4 件に関して，1 件はアクセスができないため判別がつかず，1 件はドメイン廃止通知 Web ページの一致による検知であり，2 件は事業者によるアクセス遮断通知 Web ページの一致による検知であった．PhishTank データから稼働中ではない URL を除去していたため，上記の 4 件のうちドメイン廃止通知 Web ページ及びアクセス遮断通知 Web ページの一致により検知された 3 件は本来検知されないはずであるが，PhishTank データの稼働中か否かの情報が正確ではない場合があるため，稼働中ではない Web ページの視覚的特徴の類似性を基に検知されてしまったものと考えられる．いずれにおいてもサイトの視覚的特徴が異なるようなサイトを捉えてはいないため，検知手法の動作からは想定通りの結果が得られたことになる．しかしながら，誤検知を減らすためにも上記のようなサイトを除外する仕組みの検討が必要である．

5.2.2 出現比率の評価

PhishTank データにおいて観測されたリソース統計情報 2，リソース統計情報 3 に関して，PhishTank データと実証実験データにおける出現回数を調査した．例えば PhishTank データおいて多く観測され，実証実験データにおいてほとんど観測されないなどの出現回数の偏りが見られる場合は，PhishTank データにおいて多く観測されるリソース統計情報をフィッシングサイトに特有のリソース統計情報としてブラックリスト化することで，フィッシングサイト検出の誤検知の減少が期待できる．

PhishTank データにおいて観測された全 808 種類のリソース統計情報 2 と全 887 種類のリソース統計情報 3 に対して，以下の条件を満たすリソース統計情報を抽出した．また 2019 年 5 月，6 月の実証実験データにおいて条件 P を満たすリソース統計情報を持つ URL を抽出し，その悪性判定を実施した．

条件 P

PhishTank データにおける出現回数が WarpDrive データにおける出現回数の 10 倍を超えるようなリソース統計情報

表 8 実験結果 (条件 P)

	リソース統計情報数	実証実験データにおける URL 数	フィッシングサイト数
リソース統計情報 2	3	5	1
リソース統計情報 3	4	4	1

結果を**表 8** に示す．リソース統計情報 2，リソース統計情報 3 からフィッシングサイト検知に有効と思われるリソース統計情報を大幅に絞り込むことができた．また，絞り込んだリソース統計情報により未知のフィッシングサイトを 1 件検知することができた．

以上より，PhishTank データにおける出現回数が WarpDrive データにおける出現回数を大幅に上回るようなリソース統計情報をブラックリスト化する運用には一定の効果が見込まれる．

5.2.3 既知のフィッシングサイト情報に GSB を用いる場合

表 9 リソース統計情報 1 による検知未知フィッシングサイト数 (2019 年 5 月)

	検知した未知フィッシングサイト数	リソース統計情報数
リソース統計情報 1	27	61,569

既知のフィッシングサイト情報に GSB を用いてフィッシングサイトの検知を実施した．使用したデータは 2019 年 5 月のデータであり，GSB の情報はブラウザセンサで取得した情報を用いた．尚，GSB と実証実験データの突合結果は 23URL であった．

リソース統計情報 1 により検知した未知フィッシングサイト数を**表 9** に示す．検知した未知のフィッシングサイト群に関して，検知時点で既にサイトが閉鎖する等してスクリーンショットによる比較を実施できていないが，リソース統計情報 1 により検知されていること，トップレベルドメインやクエリ文字列に偏りが見られること，及びリソース統計情報 1 を用いて検知された URL 群に関しては概ね視覚的特徴が一致しているという検証結果により，視覚的特徴は一致していたと考えられる．

6. 考察

6.1 リソース統計情報の選定

4 章，5 章より，リソース統計情報のみをフィッシングサイト検知に用いる場合には，リソース統計情報 1 が有効であることが分かった．また 5.2.2 節より、リソース統計情報 2、3 についても正規サイトでの観測数が少ないリソー

ス統計情報を用いることでリソース統計情報 1 と同程度の効果が期待できることが分かった．

6.2 Web アクセスログの情報が不完全な場合への対応

本手法はブラウザセンサから取得した情報を基に解析を実施しているため，ブラウザセンサによる情報取得に漏れがある場合はきちんと動作しない．しかしながら，ブラウザセンサ側で画像の content-length 及びリソースをきちんと取得できない事例が確認された．ブラウザセンサ側の情報取得の不具合に対応できるように本手法を改良することで検知率の向上が期待できる．例えば，リソース統計情報の完全一致ではなく，ある程度リソース統計情報の一致している URL を検知する仕組みの検討が考えられる．

6.3 誤検知の事例

リソース統計情報 1 による誤検知の事例に関して，ドメイン廃止通知 Web ページの一致や事業者によるアクセス遮断通知 Web ページの一致によるものが見られた．いずれにおいてもサイトの視覚的特徴が異なるようなサイトを捉えてはいないため想定通りの結果が得られたと言えるが，上記のようなサイトを除外する仕組みを構築することで検知率の向上が期待できる．

7. おわりに

本稿では，著者らが提案するフィッシングサイト検知手法に関して，著者らの実施している WarpDrive 実証実験ログデータ及び PhishTank に投稿されたフィッシングサイトのデータを用いて評価を行った．結果，提案手法はスクリーンショットの比較による検知手法に近い性能が得られることが分かった．提案手法は，Web アクセスログのみを利用して実装できるため，従来手法と比較して処理量が少なく，Web プロキシサーバなど様々な監視地点に実装可能である．

今後の課題として，Web アクセスログが完全に取得できなかった場合においてフィッシングサイト検知を可能にする仕組みの検討が挙げられる．例えば類似するリソース統計情報を入力として受け取った場合に類似した値を出力する関数を定義し，その関数の出力の類似性を基にフィッシングサイトを検知する方法が考えられる．また，フィルタを再検討する等してドメイン廃止通知 Web ページや事業者によるアクセス遮断通知 Web ページを検知対象から除外する仕組みの検討も今後の課題である．

謝辞 本研究成果は，国立研究開発法人情報通信研究機構（NICT）の委託研究「Web 媒介型攻撃対策技術の実用化に向けた研究開発（WarpDrive: Web-based Attack Response with Practical and Deployable Research InitiatiVE）」により得られたものである．

参考文献

[1] 中山心太，吉浦裕，“模倣コンテンツの特性に基づくフィッシング検知方式の誤検知防止，”情報処理学会論文誌，第 50 巻，第 9 号，pp.2034-2047，Sep. 2009.
[2] “「佐川急便の偽サイト」に学ぶ Android の防御法　見直す設定はたった 1 つ (1/2),” https://www.itmedia.co.jp/news/articles/1807/31/news046.html.
[3] 松田健，加藤雅彦，唐沢勇輔，丹京真一，中村智史，林憲明，加藤孝浩，“ドメインの特徴によるフィッシング詐欺サイト探索の可能性，”第 80 回全国大会講演論文集，第 2018 巻，第 1 号，pp.447-448，Mar. 2018.
[4] 原正憲，山田明，三宅優，“ブラウザ表示を利用した悪意あるサイト検知方式の提案，”情報処理学会研究報告コンピュータセキュリティ（CSEC），第 2008 巻，第 45(2008-CSEC-041) 号，pp.49-54，May. 2008.
[5] Chrome webRequest API, https://developer.chrome.com/extensions/webRequest/#type-ResourceType
[6] “Google Safe Browsing,” https://safebrowsing.google.com/.
[7] “PhishTank,” https://www.phishtank.com/.
[8] “WarpDrive,” https://warpdrive-project.jp/.
[9] 山田明，澤谷雪子，松中隆志，田中翔真，窪田歩，“ユーザ参加型の Web 媒介型サイバー攻撃対策における実証実験，”研究報告セキュリティ心理学とトラスト（SPT），第 2019-SPT-32 巻，第 9 号，pp.1-6，Feb. 2019.
[10] “WarpDrive,” https://warpdrive-project.jp/terms.html.