ドライブ・バイ・ダウンロード攻撃検知のための悪性サイト情報収集手法の改善

吉田 豊[1]　中村 嘉隆[2]　稲村 浩[2]　高橋 修[2]

概要：Drive-by-Download 攻撃は，Web の利用者の大きな脅威となっている．この攻撃は攻撃者によって改ざんされた正規サイトへユーザがアクセスすることによって生じる受動的攻撃である．近年では攻撃が複雑化しており，調査や検知技術を逃れるクローキングという手法の存在も報告されている．Drive-by-Download 攻撃を防止するためには，素早い検知と攻撃の情報収集が重要である．著者らは，悪性サイトの早期発見を目的としてクライアント環境で攻撃を行う悪性サイトの検知を行い情報を収集する手法を提案した．しかし，マルウェアのダウンロードが発生しなければ検知できない問題がある．そこで，提案手法に悪性サイトの JavaScript の特徴を利用した検知手法が適用できるか検討を行う．

An improvement of information gathering method of malignant site for detection of Drive by Download Attack

YUTAKA YOSHIDA[1]　YOSHITAKA NAKAMURA[2]　HIROSHI INAMURA[2]　OSAMU TAKAHASHI[2]

1. はじめに

近年，ドライブ・バイ・ダウンロード攻撃 (Drive-by-Download attack:以下 DbD 攻撃) がマルウェアを配布するための有力な手段となっている．この攻撃はウェブブラウザに関するソフトウェアの脆弱性を利用しており，ユーザが Web ページへアクセスするだけでマルウェアに感染してしまう．そのためマルウェアによる被害が発覚するまで感染に気が付きにくいという特徴がある．ドライブ・バイ・ダウンロード攻撃の典型的な流れを図 1 に示す．はじめに，攻撃者は正規のサーバに対して脆弱性攻撃を行うなどの手段を用いて，サーバのコントロールを得て Web ページを改ざんする．この改ざんの目的は，改ざんしたサイトを訪れたユーザを攻撃者の用意した攻撃を行うサイトへと遷移させるためである．ユーザが攻撃者によって改ざんされた正規サイトを訪れると，JavaScript・PHP などのスクリプトによって中継サイトへと遷移させられる．中継サイトでは，ブラウザフィンガープリンティングの技術を用いて，ユーザのブラウザやプラグインなどの環境情報の取得が行われる．取得された環境情報をもとに，ユーザに対して利用可能な脆弱性があれば，脆弱性攻撃が行われる．その後，ユーザはマルウェア配布サイトへと誘導され，最終的に悪意あるソフトウェアをダウンロードさせられる．

近年の DbD 攻撃では，既存の対策・調査技術を回避するクローキング技術が用いられている．例えば，攻撃を行うサイトのドメインを短期間のうちに変更することによって，ブラックリストによる対策を困難にする手法や，セキュリティベンダなどの IP アドレスからのアクセスに対して正規の振る舞いをするといった手法がある．このように複雑化する攻撃に対して，DbD 攻撃を行うサイトの早期発見・検知が非常に重要となる．攻撃を行うサイトの情報収集を目的として，クライアント環境下で攻撃の検知時に悪性サイトの情報を送信するセンサを利用した手法が提案されている [1]．この手法はクライアント環境で，悪性サイトの検知を行うセンサと，センサによって悪性と判断されたサイトの情報を格納するデータベースから構成される．センサは，Web 空間から悪性サイトを抽出する簡易フィルタとして機能し，悪性サイトの特徴を持ったサイトのデータを収

[1] 公立はこだて未来大学大学院 システム情報科学研究科
Graduate School of Systems Information Science, Future University Hakodate
[2] 公立はこだて未来大学 システム情報科学部
Systems Information Science, Future University Hakodate

集する．データベースに収集された悪性サイトに関する情報は，ハニークライアントなどのクライアントの環境下で実行するのが困難な調査に使用することができる．現在本手法は，DbD 攻撃のページ遷移の情報を利用し，攻撃の検知を行っている．マルウェアのダウンロードが発生した場合に攻撃が検知されるが，ユーザがマルウェア配布サイトへたどり着かなかった場合には攻撃を検知することができない．本稿ではこの問題を解決するために，悪性サイトの JavaScript に着目し，センサによる悪性サイトの攻撃検知可能な範囲を拡張することを目的として，JavaScript の特徴を用いた悪性サイト検知手法の検討を行う．

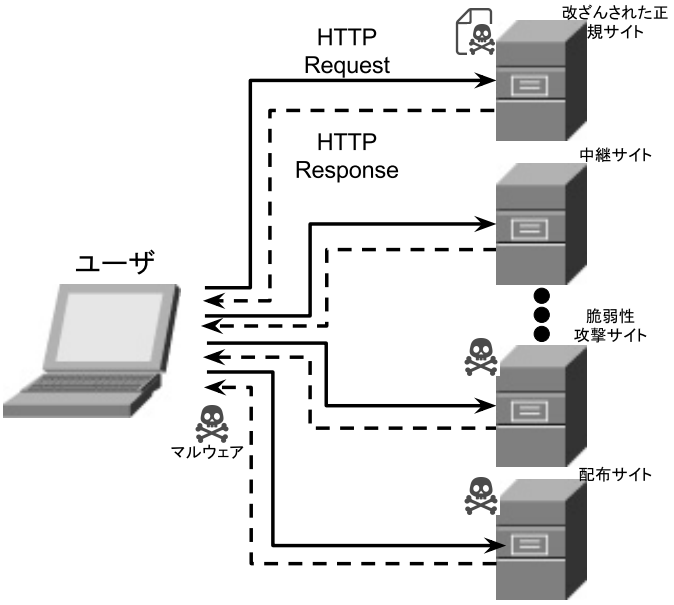

図 1　ドライブ・バイ・ダウンロード攻撃のフロー

2. 関連研究

2.1 悪性サイトの情報収集手法

改ざんされた正規サイトや実際に攻撃を行うサイトなどの DbD 攻撃に関連したサイトの情報を収集する技術に，クローラを用いたものがある．クローラは自ら Web ページへアクセスし任意の判定手法によって，訪れたサイトが悪性かどうか判断する．クローラによる調査には，いかにして調査先を決定するかという問題がある．クローラによる Web ページの探索を効率的に行うための手法として，既知の悪性サイトの特徴を用いたものがある [2]．この手法で用いられる特徴は，リンク構造，コンテンツの類似性などがある．検索エンジンでこれらの特徴を検索し，検索結果をクローリング先として利用することによって，既知の悪性サイトと関連のあるサイトを発見可能にしている．

一方，ユーザのブラウジング情報を収集し，大規模な解析センタによってブラウジング情報を分析することによって，DbD 攻撃の脅威からクライアントを守る DbD 攻撃対策フレームワークがある [3], [4], [5]．このフレームワークはユーザのブラウジングの監視を行うセンサと，センサによって得られる情報の分析を行う解析センタから構成される．センサはユーザのブラウジング情報として，アクセスした URL・ダウンロードしたコンテンツのハッシュ値・HTTP Request/Response ヘッダなどの情報を収集し，観測センタへ送信する．解析センタでは，受信した情報をもとに DbD 攻撃の有無をリアルタイムに検知する．攻撃を検知した場合には，センサに対して攻撃の発生を通知することによって，ユーザの危険なサイトへのアクセスを防止することができる．本手法では，ユーザのブラウジング情報を収集することによって，様々な検知手法を利用することができる．検知手法には，Web サイトの遷移先サイト情報を監視し，変化があった際に遷移先サイトが安全であると判断された有名サイトでなかった場合に悪性化を疑う方法などがある．

2.2 攻撃検知手法

攻撃検知手法のひとつに，DbD 攻撃の特徴的な通信遷移の情報を利用したものがある [6]．DbD 攻撃では，改ざんされた正規サイトへアクセスしたユーザがブラウザの自動読み込みやリダイレクトによって，中継サイト・脆弱性攻撃サイト・配布サイトのように遷移させられる．ブラウザによって自動的に読み込まれるファイルに対して，深さ・広がりという指標を定義しており，深さとは自動読み込みやリダイレクトの段数・広がりとは異なるドメインへのリンク跨ぎの段数としている．深さ・広がりの例を図 2 に示す．本手法では，ユーザ操作による Web サイトのアクセスを起点として，深さ・広がりを計測する．深さ・広がりの指標が設定した閾値に達する，かつ HTTP レスポンスの Content-Type がマルウェアの可能性があるものである場合に悪性と判定する．Content-Type がバイナリファイルや PDF などのマルウェアの可能性があるものを検知することによって攻撃を防ぐことができる．

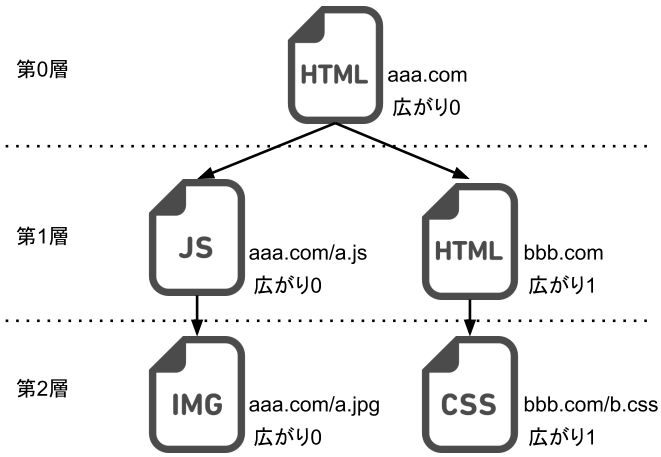

図 2　深さ・広がりの例

2.3　攻撃に使用される JavaScript

高田らによって DbD 攻撃で使用される JavaScript について調査が行われている [7]．調査には，低対話型ハニークライアントとして HtmlUnit[8] を使用しており，JavaScript の関数の実行回数，実行される JavaScript 関数の傾向などを調査している．調査によると，JavaScript 関数の回数や関数の種類の割合は難読化処理の影響によるところが大きいと報告されている．

2.4　悪性サイトの情報収集手法の問題

既知の悪性サイトの特徴を用いることによってクローラの調査効率を改善する手法は，調査のもととなる悪性サイトの情報がない場合には，効果を発揮することができない．またクローラによる悪性サイトの情報収集の最も大きな課題に，情報の鮮度の問題がある．クローラによって悪性ではないと判断されてから，次に再調査されるまでの間に改ざんが行われた場合には，DbD 攻撃による被害が発生する可能性がある．

ユーザのブラウジング情報を収集する手法では，クローラによる調査のような鮮度の問題は生じない．様々な検知手法を適用可能なため，攻撃の検知率を高めることも容易である．しかし，監視するユーザ数によって処理量が増加してしまい．データの収集や解析に多くのコストがかかるという問題がある．

3.　センサを利用した悪性サイトの情報収集手法

3.1　概要

著者らは，DbD 攻撃の早期発見・検知を目的とした，クライアント環境下で攻撃の検知時に攻撃サイトの情報を送信するセンサを利用した手法を提案している [1]．本手法の概要図を図 3 に示す．本手法は，クライアント環境で DbD 攻撃の検知し，検知したサイトの情報を送信するセンサとセンサから送られる情報を集積するデータベースから構成されている．

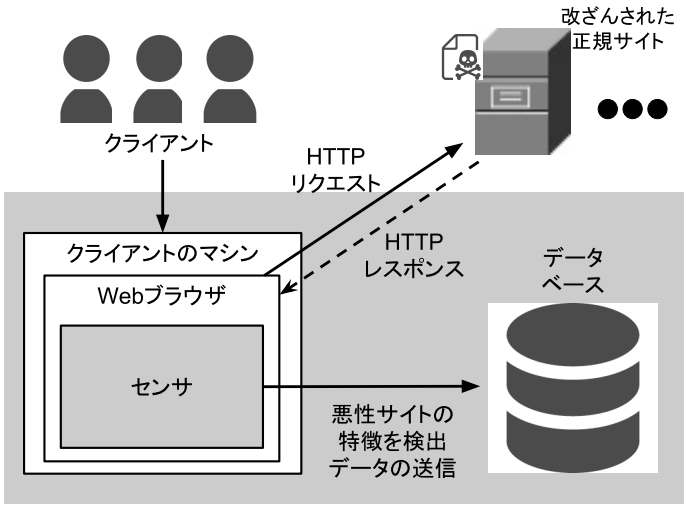

図 3　提案手法概要

センサはユーザの Web へのアクセスを監視し，Web 空間から悪性サイトを抽出するような働きをする．悪性サイトが正規サイトよりも高確率で持つ特徴を検知し，データベースへ収集することによって，センサは Web 空間から悪性の特徴をもつサイトを集めることができる．このようにして収集されたデータの中には，高い確率で悪性サイトが含まれている．収集されたデータはあくまで，悪性サイトの特徴を持つサイトなので，正確に正規か悪性かを判定するためには，より詳細な調査が必要となる．この調査には，ハニークライアントなどの既存の悪性サイトの検知技術が用いられる．また，センサによって収集されるデータにはクライアントのブラウザに関する情報も含まれており，攻撃の分析に活用することができる．さらに本手法は，ユーザの Web へのアクセスの度に悪性かどうか判定を行うため，2.4 節で述べた鮮度の問題が生じない．

3.2　システム構成

センサ

センサは Web ブラウザのプラグインとして動作し，クライアントの Web ブラウジングを監視し，Web 階層のカウントを行う．マルウェアの可能性がある危険なコンテンツのダウンロードがあった場合，Web 階層の指標を用いることによって悪性かどうかの判別を行なう．

データベース

データベースはセンサから送られてくる情報を保存する．データベースには，例えば HTTP リクエスト・レスポンスの場合，文字列として格納され，Web 階層の情報は，リンクの深さ・広がりの数値が格納される．集積された情報を元にさらなる分析を加えることによって，悪性サイト

の特定にも応用できる．

3.3 収集するデータ

悪性サイトの情報として，自動遷移の始まりから，悪性と判断されたサイトを含むサイトまでの一連の遷移の情報を収集する．収集する項目は URL，HTTP リクエスト，HTTP レスポンス，アクセス時刻，Web 階層の情報，検知された悪性サイトの特徴などがある．

悪性サイトの情報だけではなく，攻撃が行われた環境情報を得るために，ユーザのブラウザの環境情報を収集する．収集する項目は，IP アドレス，OS の種類，ブラウザの種類・バージョン，ブラウザのプラグインの情報などがある．

3.4 Web 階層

本稿では，ユーザが任意のページを開いた際に，そのページを起点としてブラウザによって自動で読み込まれるページの構造を Web 階層と定義する．Web 階層の深さをあるページから自動で読み込まれたページの構造の深さをカウントしたもの．Web 階層の広がりを異なるドメインへの遷移の数と定義する．Web 階層の深さは HTTP ヘッダに含まれる Referer，Location から判断する．Web 階層の広がりは，HTTP のリクエストラインに含まれる Request-URL や HTTP ヘッダの Host を用いて判断する．

3.5 危険なコンテンツ

HTTP 通信によってダウンロードされるファイルの種類は，HTTP ヘッダの Content-Type によって判別される．マルウェアのファイルの種類には，PDF ファイル，SWF ファイル，バイナリファイルなどがある．マルウェアの可能性のある Content-Type の一例を表 1 に示す．

表 1 危険な Content-Type

Content-Type
appplication/pdf
application/x-download
application/x-msdownload
application/x-msdos-program
application/octet-stream
application/x-shockwave-flash

3.6 悪性サイトの検知手法

センサでは悪性サイトを検知するために，Web 階層の特徴を利用している．ユーザ操作による Web アクセスを起点として，リンクの深さ・広がりを 0 とする．ブラウザによる自動通信の間，HTTP リクエスト・レスポンスのヘッダの情報である Referer・Location から深さ・広がりをカウントする．深さが 2 以上または，広がりが 1 以上のとき，マルウェアの可能性のある Content-Type をもつファイルをダウンロードした場合，悪性と判定する．

収集されたデータの利用者は，URL や Conent-Type などデータベースに格納されている情報を用いることによって，特定の Conent-Type を持つリクエスト等の調査が可能となり，既存のマルウェアの検知手法を効果的に活用することができる．

4. 実験・評価

センサを利用した悪性サイトの情報収集手法における攻撃の検知手法の有効性を評価するため実験を行った．実験用データとして良性通信データ・悪性通信データを用いて，それぞれの誤検知率・検知率から評価を行う．

4.1 実験に使用する通信データ

良性通信データとして，Alexa[9] がランキングしているトップ 1000000 の中からランダムにアクセスしたものを用いた．Alexa に登録されているサイトは，必ずしも正規サイトである保証はないものの，正規サイトであるという仮定で実験を行った．Web サイトへのアクセスには，ヘッドレスブラウザである PhantomJS[10] を利用した．通信パケットの収集には，tcpdump を使用した．

悪性通信データとして，NTT セキュアプラットフォーム研究所が収集した D3M データセット 2014 を使用した．D3M(Drive-by-Download Data by Marionette) は，高対話型の Web クライアント型ハニーポットで収集した，攻撃通信データ，マルウェア検体，マルウェア検体の通信データが含まれる [11]．D3M では，攻撃検知した URL へ直接アクセスしているため，DbD 攻撃の起点となる改ざんされた正規サイトに関する情報は含まれていない．また，今回の実験ではマルウェアの配布まで行われていない通信データは，本検知手法の適用範囲外なので除外した．

4.2 実験方法

良性・悪性データのパケットから通信の再現に必要となる情報を抽出し，HTTP リクエスト・レスポンスを解析することによって，行われた通信を再現した．パケットから抽出した項目は，HTTP リクエストでは Referer・Host・リクエスト URI・送信元ポート番号・送信先ポート番号などがある．HTTP レスポンスでは，Location・Content-Type・送信元ポート番号・送信先ポート番号などを抽出した．抽出したデータを用いて，再現された通信に対して 3.6 節の手法を適用することによって，誤検知率・検知率を求めた．

4.3 結果・評価

実験の結果を表 2 に示す．良性の有効 URL 数が 268 個，アクセスした URL 単位の正規サイトの誤検知数が 51 個であったため，誤検知率は 19%となった．HTTP セッション数で見ると，総セッション数が 21827 で誤検知数が 138 個

であるため，0.6%となった．

悪性データについては，有効 URL 数が 49 個であり，すべての URL で攻撃を検知することができた．

良性データの誤検知の要因を確認してみると，要因のひとつとしてウェブフォントファイルをダウンロードする際に，Content-Type が octet-stream である woff ファイルがあった．octet-stream は，バイナリファイルであり，攻撃成功時に現れるレスポンスのひとつである．一連の通信遷移の後に octet-stream へのアクセスが行われたため，誤検知が発生した．

表 2 実験結果

項目	スロット数	割合 (%)	HTTP セッション数	割合 (%)
良性有効 URL 数	268	-	21827	-
誤検知	51	19.0	138	0.6
悪性有効 URL 数	49	-	-	-
検知	49	100	-	-

4.4 考察

本検知手法は悪性サイトの見落としが少なく，かつ良性サイトを悪性と誤検知してしまうことも少なかった．この要因として考えられるのは，PDF やバイナリといった危険なファイルに対するリクエストを検知のトリガとしたことが挙げられる．また，一般的な良性サイトにそのようなリクエストが含まれていることが多くないということもひとつの要因であると考えられる．

5. センサの検知手法の拡張

DbD 攻撃では，ユーザの環境情報を取得し，対象に脆弱性があるかを確認してから攻撃を行う．ユーザに脆弱性が無い場合には，攻撃は行われず Google などのサイトにリダイレクトすることがある．このように，DbD 攻撃を行うサイトは訪れたユーザ全てに対して攻撃を行なうのではなく，攻撃可能な対象を見極めて攻撃を行う．単にマルウェアのダウンロードをトリガとした検知手法は，悪性サイトへアクセスしただけでは攻撃の検知ができないという問題がある．

DbD 攻撃のマルウェアの配布以外の特徴として，悪性な JavaScript の実行が挙げられる．DbD 攻撃で使用される JavaScript には，難読化処理やブラウザフィンガープリンティングなどの特徴が挙げられる．本稿では，悪性サイトの JavaScript に着目し，悪性サイトの検知に利用可能かどうか検討する．

5.1 難読化処理

悪性サイトの JavaScript には，解析を避けることを目的として難読化処理がされている場合がある．難読化の解除のために，substr()・concat() などの String 関数と createElement()・write() などの DOM 関数が使用される．また，難読化の解除には多くの関数呼び出しが行われるため，関数の実行数が多いという特徴がある．このため String 関数・DOM 関数が多く実行されている場合，難読化処理が行われていると予想できる．しかし正規サイトでも，プログラムの保護を目的として難読化処理が行われる場合があるため，単純に String 関数・DOM 関数の実行数で悪性サイトであると判定することは難しい．

一方，難読化処理の検知後にマルウェアの可能性のあるコンテンツへのアクセスがあった場合には攻撃の可能性が高まるが，結局攻撃の発生をトリガとしなければならないので，難読化処理の特徴の検知手法は提案手法における攻撃検知範囲の拡張という目的にはそぐわないと考えられる．

5.2 ブラウザフィンガープリンティング

攻撃の成功確率を高めるために，攻撃を行う前にユーザのブラウザの環境情報を取得する場合がある．この技術はブラウザフィンガープリンティングと呼ばれている．ブラウザフィンガープリンティングでは，攻撃対象の PDF・Java・ActiveX などの環境情報の取得が行われる．また，ブラウザフィンガープリンティングに用いられるライブラリとして，PluginDetect がある [12]．PluginDetect では，Flash や PDF など様々なプラグインに関する情報を確認可能である．また PluginDetect は，いくつかの Exploit Kit での利用が確認されている．ブラウザフィンガープリンティングに用いられる実装は，PluginDetect 以外にも，Exploit Kit 独自の実装も報告されており，単に PluginDetect の有無のみでは悪性・良性の判断として不十分である．

5.3 JavaScript 関数の実行数

5.1 節で述べたように，難読化処理は JavaScript 関数の実行数に大きく影響している．高田らの調査によると，JavaScript 関数の実行回数の平均値は，正規のサイトで 3,173.2 回で，悪性サイトで 90,901.2 回となっている [7]．また悪性サイトの標準偏差が 47,324.3 と大きな値となっており，正規サイト・悪性サイトで共に回数が重なり合っている箇所がある．しかし，両者の実行回数の最大値には大きな開きがあるため，一部の実行回数の多いものについては悪性であると判断できる可能性がある．

5.4 複数の特徴を用いた検知

難読化処理のように，正規サイト・悪性サイト共に用いられているものを単体で悪性サイト検知の特徴として利用するのは困難である．しかしながら，難読化処理やブラウザフィンガープリンティングなどの DbD 攻撃の特徴が同時に現れた場合には，DbD 攻撃の可能性が高まる．特に DbD 攻撃の最も大きな特徴であるページ遷移と悪性サイトの JavaScript の特徴が同時に検知された場合には，攻撃の挙動に近いものとなる．また，改ざんされた正規サイト

から遷移し，攻撃に利用される JavaScript の特徴が現れるという挙動が，DbD 攻撃独特のものであれば，マルウェアのダウンロードが起こる前に攻撃検知が可能であると考えられる．

6. おわりに

本稿では，センサを利用した悪性サイトの情報収集手法がマルウェアのダウンロードに依存しているという問題を解決するために，悪性サイトの JavaScript に着目し，マルウェアのダウンロードに依存しない悪性サイトの検知手法について検討を行った．検討の結果，DbD 攻撃の大きな特徴であるページ遷移の情報と悪性サイトの JavaScript の特徴という複数の観点を取り入れることによって悪性サイトの検知性能の向上の可能性が示された．今後は，複数の特徴を用いた検知手法について実験を行い，有効性の調査を行う．

参考文献

[1] 吉田 豊, 中村 嘉隆, 高橋 修. クライアント環境で動作するセンサを利用したドライブ・バイ・ダウンロード攻撃の検出手法の提案. **情報処理学会第 *78* 回全国大会講演論文集**, 2016.

[2] Invernizzi Luca, Comparetti Paolo Milani, Benvenuti Stefano, Kruegel Christopher, Cova Marco, Vigna Giovanni. EvilSeed: A guided approach to finding malicious web pages. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, 428-442, 2012.

[3] 笠間 貴弘, 井上 大介, 衛藤 将史, 中里 純二, 中尾康二. ドライブ・バイ・ダウンロード攻撃対策フレームワークの提案. **コンピュータセキュリティシンポジウム *2011* 論文集**, 2011(3):780–785, 2011.

[4] 松中 隆志, 山田 明, 窪田 歩. Drive-by Download 攻撃対策フレームワーク実現に向けたリンク構造解析による Web サイトの分析. **情報処理学会研究報告**, 2015-CSEC-68(48), 1-8, 2015.

[5] Takashi Matsunaka, Ayumu Kubota, Takahiro Kasama. An Approach to Detect Drive-By Download by Observing the Web Page Transition Behaviors. *In Proceedings of the 2014 Ninth Asia Joint Conference on Information Security*, 19-25, 2014.

[6] 安藤 槙悟, 寺田 真敏, 菊池 浩明, 趙 晋輝. 通信の遷移に着目した不正リダイレクトの検出による悪性 Web サイト検知システムの提案. **電子情報通信学会技術研究報告**, 111(123):205–210.

[7] 高田 雄太, 秋山 満昭, 針生 剛男. ドライブバイダウンロード攻撃に使用される悪性な JavaScript の実態調査. **情報処理学会研究報告**, 2014-SPT-8.(11):1-6, 2014.

[8] Gargoyle software inc.. ”htmlunit”. http://htmlunit.sourceforge.net/.

[9] Alexa. http://www.alexa.com/.

[10] Phantomjs. http://phantomjs.org.

[11] 秋山 満昭, 神薗 雅紀, 松木 隆宏. マルウェア対策のための研究用データセット：Mws datasets 2014 (情報セキュリティ). **電子情報通信学会技術研究報告**, 115(114):125–131, 2014.

[12] Plugindetect. http://www.pinlady.net/PluginDetect.