Computer Security Symposium 2017
23 - 25 October 2017

悪性URLの強調表示による Drive-by Download攻撃解析支援手法の提案

尾崎 幸也[1]　上山 真也[1]　小西 達也[1]　山崎 雅斗[1]　坂東 翼[2]　小林 孝史[1]

概要：今日では，攻撃者の身元偽装や JavaScript の難読化により Web アプリケーションへの攻撃の解析は困難になっている．先行研究にて，我々は Drive-by Download 攻撃によって生じた多段のリダイレクト系列を木構造として視覚化するアプリケーションを開発した．本研究ではこれに，URL 文字列に対する決定木分析を元に得られた悪性の特徴量をもつ URL を強調表示する機能を付与する．悪性の URL を強調表示することで，さらに効率的な解析支援を目指す．

キーワード：MWS，Drive-by Download 攻撃，攻撃の可視化，決定木

Proposal to support analysis of Drive-by Download attack by highlighting malicious URL

Koya Ozaki[1]　Shinya Ueyama[1]　Tatsuya　Konishi[1]　Masato Yamazaki[1]　Tsubasa Bando[2]
Takashi Kobayashi[1]

Abstract: Nowadays, it is difficult to analyze attacks on web applications due to the attacker's identity camouflage and the obfuscation of JavaScript. So far, we have developed an application to visualize the multistage redirecting caused by Drive-by Download attacks, as a tree. In this research, we add a function to highlight URLs with malignant quantities obtained by decision tree learning. We aim support for a more efficient analysis to by highlighting malignant URLs.

Keywords: MWS, Drive-by Download attack, Attack visualization, Decision tree

1. はじめに

Web アプリケーションの脆弱性を悪用した攻撃は多様化し，さらにエクスプロイトキットなどの入手が簡単化したことで手軽に攻撃が可能になってしまったことから，被害は拡大している [1]．また，攻撃者は Web アプリケーションそのものに攻撃を仕掛けるだけでなく，対象の Web アプリケーションを改ざんする．その後，接続したユーザに複数の URL を経由させ，ブラウザやプラグインの脆弱性を悪用し，ユーザには秘密裏にマルウェアをダウンロードさせる Drive-by Download 攻撃の被害も増加している．

この攻撃手法は，ユーザのブラウザに複数のサイトへの多段階のアクセスを強制させるため，リダイレクト系列を作る特徴を持つ．これらすべての URL を解析するには高い専門性と膨大な時間が必要になるため，実際にマルウェアをダウンロードさせるサーバの特定に非常に時間がかかる．

青山らの研究 [2] では，通信記録から取り出したパケット及び，実際にクローリングすることで収集したパケットを木構造で視覚化した．本研究では，木構造化した各 URL を解析し，決定木分析によって URL の良悪判定を行い，悪性 URL を色付けし強調することで解析の支援を行うアプリケーションを開発した．パケットの解析及び，悪性 URL の検知までを本アプリケーションを用いて自動化すること

[1] 関西大学
Kansai University
[2] 関西大学大学院
Kansai University Graduate School

により，より効率的な解析支援をめざす．

2. 提案手法

2.1 システム概要

本提案では，ユーザのコンテンツ利用により発生したパケットを解析し，その中で Drive-by Download 攻撃の発生源，あるいはリダイレクト先などの URL を色付けし視覚化するシステムを提案する．

悪性 URL から特徴量を抽出し，機械学習を用いて判定する方法は実施されてきている．本提案でも従来から用いられている特徴量を抽出し，決定木による判定を行う．

2.2 URL の特徴抽出

URL の特徴量としては，様々な提案がなされている．本提案で抽出した特徴を表 1 に表す．

表 1 ベクトル成分
Table 1 Vector component

番号	ベクトルの成分要素
(1)	URL の全長
(2)	特殊記号が含まれる数
(3)	パス文字列長
(4)	クエリ文字列長
(5)	文字列が含まれる数
(6)	クエリに含まれる key の数
(7)	数字が含まれる数
(8)	URL に含まれるポート番号
(9)	TLD
(10)	URL 中の IP アドレスの有無

(1)，(2) については，L. Xu の研究 [3] で述べられている悪性 URL はその全長が平均して長くなるという点と，悪性 URL に含まれる特殊記号は多くなるという点を元にしている．(3)〜(6) については，佐藤らの研究 [4] で述べられている悪性の URL の全長が長くなると，パス，クエリ，URL 中の文字列も長くなる点と，key の数が増加することにより，クエリ文字列長も長くなる可能性が高く，また比例して全体の文字列長も長くなるという点を元に特徴量とした．(7) については J. Ma らの研究 [5] で判定の要素として選定されている．(8) については，URL に含まれるポートに制限はないため，接続されるポート番号に特徴が出ると考えられるので，ポート番号の有無ではなく，ポート番号を特徴量とした．(9)，(10) については山西らの研究 [6] で判定の要素として選定されている．

2.3 決定木による判定

本提案によるシステムを図 1 に示す．ユーザによるクローリングないし pcap ファイルから収集された各 URL をサーバ側へと送信する．サーバ側では受け取った URL から表 1 で示した特徴量を抽出し，それらをベクトル表現に変換する．例えば，`http://www.hoge.com/test/?a=b&key=value` という URL を変換した場合，[58,18,6,30,29,2,11,80,0,1] となる．そして，そのベクトルを用いて決定木による判定を行う．判定結果は悪性である場合 1，良性である場合は 0 としその結果をユーザ側に返す．

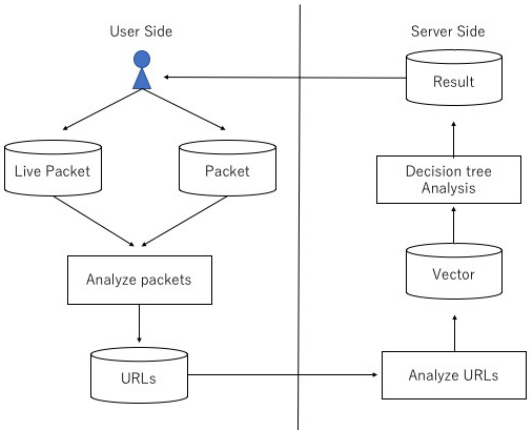

図 1 システムの流れ
Fig. 1 System flow

このように，サーバ・クライアント型を採用している理由としては，複数の解析者による同時利用を考慮したためである．同時にアプリケーションを利用した場合でも，検知結果などを一箇所に蓄積することで，データ管理の利便性の向上が期待できる．

2.4 木構造と強調表示による視覚化

トラフィックデータから TCP パケットを抽出し，JavaScript の実行などの解析からセッションの関連性を解析し，その結果を元にトラフィックデータ内の HTTP セッションを木構造として視覚化する．こうして視覚化することで，Web ページの変遷を明確にすることができる．

トラフィックデータは pcap ファイルだけでなく，ユーザによるクローリングでも収集することができ，解析者が実際に Web ページを閲覧した履歴に対しても解析を行うことが可能である．

こうして木構造化したのち，サーバ側から判定された結果を元に悪性の URL を赤く色付けすることで，強調表示を行う．表示イメージを図 2 に示す．

3. 関連研究

3.1 リダイレクト系列の木構造化に関する研究

Drive-by Download 攻撃は攻撃対象の絞り込みを行うため多段のリダイレクト系列を作る．青山らの研究 [2] では，パケットの解析，及び JavaScript の実行によって作られたリダイレクト系列を木構造化し視覚化する．そうすることで攻撃の発生位置の特定や，用いられたサーバの特定といった解析作業の困難性を低減するアプリケーションが開

図 2 強調表示の例
Fig. 2 Examples of highlighting

発された．

本研究では，木構造による視覚化だけでなく，実際に悪性だと考えられる URL を検知しそれらを強調表示することで，解析者の支援を行う．

3.2 機械学習による Drive-by Download 攻撃検知に関する研究

Drive-by Download 攻撃により生じたリダイレクト系列や URL から特徴を見つけ出し，機械学習によって検知する手法が盛んに提案されている．佐藤らの研究 [4] では，URL の文字長や URL に含まれる特殊記号の数といった URL の特徴量を元に決定木による判定を行い，悪性 URL の検知を行う手法が提案されている．

山西らの研究 [6] では，URL とドメインの特徴だけでなく隣り合った URL 系列の特徴から，畳み込みニューラルネットワークを拡張した Event De-noising Convolutional Neural Network を用いて Drive-by Download 攻撃の検知を行う手法が提案されている．

本研究では，URL の長さや TLD，ポート番号といった特徴量を用いて決定木を作成し，それによって検知した結果を強調表示によって視覚化する．

4. 実装・評価

本システムはユーザ側，サーバ側で実装形態が異なっており，ユーザ側は JavaSE version 1.8，サーバ側は Python version 3.5 を用いて実装した．また，主要なライブラリとして，ユーザ側はトラフィックデータのキャプチャに jNetPcap version 1.3.0，JavaScript の実行に HtmlUnit version 2.19，HTML のパースに Jericho HTML Parser version 3.4 を使用した．加えて jNetPcap の実行に WinPcap version 4.1.3 を使用した．サーバ側は，取得した URL 文字列の解析に urllib3 version 1.13，決定木による判定に scikit-learn version 0.18 を使用した．

4.1 HTTP セッションの関連性の解析

本システムでは，トラフィックデータを解析し，HTTP セッションの関連性を明らかにするために，青山らが開発したアプリケーション [2] を元に開発する．これによって抽出された HTTP セッションから取り出された URL 文字列に対して良悪判定を行う．

4.1.1 HTTP セッションの抽出

収集したトラフィックデータ内に含まれる HTTP リクエストパケット及び，HTTP レスポンスパケットを抽出することで HTTP セッションの抽出を行う．はじめに TCP パケットを参照し，その中に HTTP リクエストパケットが存在した場合，そのパケットを暫定的に HTTP パケットの始点であるとする．その後，HTTP リクエストパケットに対し ACK パケットを受信していた場合は一つの HTTP セッションとする．PSH フラグを含むパケットを受信した場合はそのパケットを暫定的な HTTP セッションの終点とする．同一のセッション内に FIN フラグを確認できなかった場合は，暫定的な終点を HTTP セッションの終点とする．

4.1.2 関連性の解析

HTTP セッションに含まれる HTTP リクエストパケット及び HTTP レスポンスパケットに含まれる情報を元に，HTTP セッションの関連性を解析する．

HTTP リクエストパケットのうち，解析に用いる要素を表 2 に示す．

表 2 リクエストパケットの要素
Table 2 HTTP Request Header

要素	内容
Request URI	サーバに対する要求の種類
Request Method	要求の対象であるファイルパス
Request Version	通信で用いる HTTP のバージョン
Referer	対象 Web ページのリンク元の Web ページ

HTTP レスポンスパケットのうち，解析に用いる要素を表 3 に示す．

表 3 レスポンスパケットの要素
Table 3 HTTP Response Header

要素	内容
HTTP Version	通信で用いられる HTTP のバージョン
Status Code	要求に対する応答番号
Status Message	Status Code の略称
Location	転送先の URL
Message part	Web サーバから送信される HTML データ等

HTTP セッションの関連性を解析する際，Referer や Location を用いる．しかし，これらは HTTP パケットに記載されていない場合があり，その場合は関連性を明らか

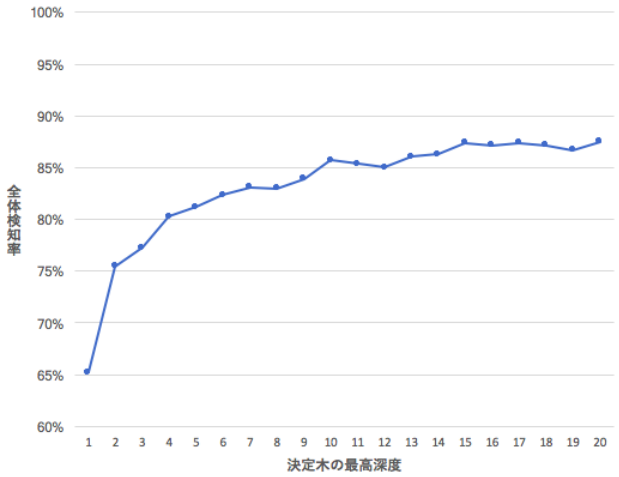

図 3 深度に対する検知率のグラフ
Fig. 3 Graph of detection rate against depth

にすることができない．よって，Referer や Location だけでなく，表 2，表 3 で示した Referer や Location 以外の要素も含めて解析を行う．

4.2 URL の良悪判定

URL の良悪判定は，URL 文字列の静的解析によって特徴量をベクトルで表現したのち，そのベクトルを用いた決定木分析によって行われる．そのため，まずユーザ側で収集した URL 文字列を解析し，表 1 で示した特徴量を抽出し，それらをベクトル化する．その後ベクトル化した特徴量を決定木によって 2 値分類する．決定木の作成には悪性データとして D3M（Drive-by Download Dataset by Marionette）[7] の 2010 から 2015 の URL リスト，良性データとして有志によって収集され，接続した際に悪性の動作が見受けられなかった URL 群を用いた．これらの URL 群を悪性は 1，良性は 0 とラベリングし学習を行なった．

また決定木の最高深度は，深度を 1 ずつ深くし，その時の全体検知率を確認することで，その点から深度を深くしても検知率の上昇が見受けられなかった時点を最高深度とした．各深度に対する全体検知率を図 3 に示す．本システムの最高深度は全体検知率の上昇が見受けられなくなった 15 とした．

この学習結果を用いて分類された結果をユーザ側に返し，受け取ったユーザ側は悪性だと判断された表示列の先頭にマーカーをつけることで色付けの判断を行う．その後木構造を作成し，マーカーのついた表示列は赤くすることで強調表示する．

4.3 評価方法

本システムの評価は，ランダムに良性データ 420 件と悪性データをそれぞれ 420 件ずつ抜き出し，それらを学習データとして使用せずにテストデータとすることで行う．この URL の数は，学習データ全体の 3 割に当たる．評価指標は，良性データを悪性だと判定した割合，悪性データを良性だと判定した割合，全体の検知率の 3 点とする．テストではテストデータで検知を行い，実際のラベリングと比較することで評価した．

また，URL 系列を解析した地点から，本システムで行なっている URL 文字列の解析及び，決定木による検知を行い，その結果を表示するまでの時間を計測する．計測には D3M のデータセットの内，10 件の pcap ファイルを用いる．同様に従来のシステムでも URL 系列を解析した地点から表示までの時間を計測し，本システムと比較する．

5. 結果・考察

5.1 結果

本研究で作成した決定木を用いて評価した結果，良性データを悪性だと判定した割合は 12.35%，悪性データを良性だと判定した割合は 12.87%，全体の検知率は 87.38%となった．この決定木を作成する際に用いたベクトルの中で，良性と悪性が同じものになった数は 372 件であった．これは学習データ全体の 13.30%に当たり，false negaive，false positive の割合とほぼ一致している．

また，解析結果の表示時間の比較結果を表 4 に示す．時間の単位はミリ秒とする．結果からわかるように，本システムでは解析及び，検知の動作が増えたため，実行時間は増加した．

表 4 解析結果の表示時間の比較
Table 4 The comparison of displaying time of analysis result

データセット	従来のシステム	本システム
A	149	6639
B	120	4241
C	41	1230
D	35	1202
E	37	1845
F	17	1070
G	12	305
H	12	322
I	13	566
J	14	623
K	29	1338
L	12	207

さらに，決定木を用いて D3M データセットの解析を行い，実際に強調表示した画面の一部を図 4 に示す．画像ファイルなども強調表示されている理由としては，決定木を作成した際に利用した URL 群と同様のドメインやパスをもつため，非常に類似している点が考えられる．しかし，これは悪性な URL への接続によって表示されたコンテンツである可能性が高く，そのコンテンツも悪性である可能

図 4 悪性と推定される URL リストの強調表示の例

Fig. 4 Examples of highlighting the URL lists which are guessed as malicious contents

性が非常に高いので，強調表示を行っても問題ないと考えられる．しかし，Java.com などの正規サイトである可能性が高いドメインに対してはこれに当てはまらず，過検知の恐れがあるため対応しなければならない．

5.2 考察

5.2.1 先行研究との比較

本システムの元となったアプリケーションに比べ，本システムでは新たに URL 文字列での検知機能を取り入れた．これにより，先行研究では攻撃箇所の最終的な発見は全て解析者に一任されていたが，本システムでは入り口サイトや，そこに含まれるコンテンツを強調することでそれらの発見を支援することができると考えられる．

また，解析の実行時間も増加しているが，最大でも 6490 ミリ秒であり，解析の妨げになるほど大きな差ではないと考えられる．しかし，あまりにも膨大なパケットを解析する場合，その大きさに比例して解析時間が増加する可能性がある．そのため，解析及び，検知の高速化が必要になる．

5.2.2 精度向上への対策

本提案では，D3M に収録されている URL 文字列の特徴を元に決定木を作成しているため，それらに類似した特徴をもつ URL 文字列にはある程度の精度を持つ．しかし，収録されていない特徴をもつ URL 文字列だった場合，精度は落ちてしまうと考えられる．この問題を回避するために，D3M に収録されている URL 文字列だけを利用するのではなく，他の攻撃パケットに含まれた URL も学習に用いることが考えられる．

また URL 文字列に含まれる特徴だけでなく，Drive-by Download 攻撃の動的な特徴や URL の系列に含まれる特徴も解析し，多角的な視点から学習をおこなうことが考えられる．多角的に学習を行うことで，悪性ベクトルと良性ベクトルが一致してしまう可能性を抑えることができると考えられる．

5.2.3 HTTPS を用いた攻撃への対策

現在，本提案は HTTPS などの暗号通信を絡めた攻撃は検知できない．理由としては，検知を行う前に URL を収集する段階でキャプチャした通信を復号できないからである．しかし，この HTTPS を用いた攻撃が増加しており，対策しなければならない．そこで我々は，本提案を MITM Proxy（Man in the Middle Proxy）としての運用を検討している．そうすることで，暗号化された通信を復号することができ，検知を行うことができると考えられる．また，MITM Proxy は中間者攻撃に利用されることも考えられるが，本システムは攻撃者による利用を目的とするものではなく，解析者による利用を目的としているため，問題はないと言える．

5.2.4 実利用での評価

本システムは解析支援を目的とし開発を行った．よって，

実際にどれほどの効率化をできるのかという評価を行う必要がある．そこで本システムを一般に公開することで，多くのユーザに利用してもらいデータの収集を行うことを考えている．それらから得られたデータからさらなる効率化を目指す．

また奥田らの研究 [8] では，Drive-by Download 攻撃を再現する手法が提案されている．この手法を用いることで擬似的な攻撃を再現することができ，本システムを用いた場合と用いなかった場合の解析効率の比較を行うことができると考えられる．

6. まとめ

我々は，Drive-by Download 攻撃によって生じた多段の URL 系列を木構造によって視覚化し，その中から悪性の特徴をもつ URL 文字列を強調表示することで，攻撃解析の支援を行うシステムを提案した．本提案では URL 文字列に含まれる特徴を元に，悪性 URL の検知に用いる決定木を作成し，良性データを悪性だと判定した割合は 12.35%，悪性データを良性だと判定した割合は 12.87%，全体検知率は 87.38%であった．また実際に URL 系列を木構造として視覚化し，作成した決定木を元に悪性 URL の強調表示をおこなった．D3M データセットの解析時間は，従来システムに比べ，最大で 6490 ミリ秒増加した．

今後は精度向上のため，他の Drive-by Download 攻撃パケットに含まれる URL 文字列による学習や，Drive-by Download 攻撃の動的な特徴量の抽出を行わなければいけない．加えて，Man in the Middle Proxy としての運用を行うことにより，HTTPS 通信にも対応する必要がある．また，実際の利用によるデータ収集を行わなければならない．

参考文献

[1] IBM：Tokyo SOC 情報分析レポート，http://www-935.ibm.com/services/jp/ja/it-services/soc-report/，(2017.08.24).

[2] 青山 佑平，吉井 章，大倉 佳歩，尾崎 幸也，坂東 翼，小林 孝史：Drive-by Download 攻撃の解析支援アプリケーションの開発と評価，Computer Security Symposium 2016，pp.819-825，2016.

[3] L. Xu, Z. Zhan, S. Xu, and K. Ye. :Cross-layer detection of malicious websites, Proceedings of the third ACM conference on Data and application security and privacy(CODASPY'13), pp.141-152, 2013.

[4] 佐藤祐磨，中村嘉隆，高橋修：エクスプロイトキットで利用される文字列特徴を用いた悪性 URL 検出手法の提案，情報処理学会研究報告 2016-CSEC-72，pp.1-6, 2016.

[5] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker.:Beyond blacklists: learning to detect malicious web sites from suspicious urls, Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining(KDD'09), pp.1245-1254, 2009.

[6] 山西 宏平，芝原 俊樹，高田 雄太，千葉 大紀，秋山 満昭，八木 毅，大下 裕一，村田 正幸：畳み込みニューラルネットワークを用いた URL 系列に基づくドライブバイダウンロード攻撃検知，Computer Security Symposium 2016，pp.811-818，2016.

[7] 高田 雄太，寺田 真敏，村上 純一，笠間 貴弘，吉岡 克成，畑田 光弘：マルウェア対策のための研究用データセット～MWS Datasets 2016～，情報処理学会研究報告 2016-CSEC-74，pp.1-8，2016.

[8] 奥田 祐樹，福田 洋治，白石 善明，井口 信和：ドライブ・バイ・ダウンロード攻撃によるインシデントを再現するフォレンジック支援システム，信学技報 ICSS2017-17，pp.81-86，2017.