Computer Security Symposium 2019
21 - 24 October 2019

悪意あるWebブラウザ拡張機能検出の試み: 良性/悪性の挙動の差に基づくデータを用いて

大石雄大[1,a)]　高田哲司[1]

概要：Web ブラウザには「拡張機能」と呼ばれるプログラムをインストールすることができ，Web ブラウザの機能を拡張することが可能である．しかし，その拡張機能にも悪意を持ったプログラムが存在し，Web ブラウザ利用者や Web サービス提供者, Web 広告事業者に様々な不利益をもたらしている．したがって，これらの悪意を持った拡張機能を検出できるようにすることが望ましい．本研究では，悪意を持った拡張機能と悪意のない拡張機能の挙動には差があるという仮説を立て，その仮説の検証を試みた．本研究ではその検証を可能にすると考える拡張機能の挙動情報を，拡張機能として実装したシステムを用いて動的解析により情報収集し，収集した挙動情報を挙動の差を表す値に変換した．この変換したデータをランダムフォレストで学習させ，悪意を持った拡張機能の検出器を実現した．小規模なデータセットに対してではあるが，この検出器の検証を試みた結果，(1) その検出器の偽陽性は 30.2%, 偽陰性は 22.2%であり，(2)Web ページに DOM 要素の追加を行う挙動を，悪意のない拡張機能と比較することが MBE 識別に有益であることが示唆された．

キーワード：Web セキュリティ，Web ブラウザ拡張機能，悪性検出，機械学習

A challenge of detecting malicious web browser extensions: using a behavioral difference between benign and malicious extensions

Kazuhiro OISHI[1,a)]　Tetsuji TAKADA[1]

Abstract: Web browser extensions are widely used in web browser users. On the other hand, it has been used as a malware by attackers. We consider that a detection system is indispensable as a countermeasure. We, then, tried to build a detection system of malicious web browser extensions using machine learning. In this paper, we report on our approach for feature extraction and the result of the preliminary experiment.

Keywords: Malware, Web security，Machine learning

1. はじめに

Web ブラウザには「拡張機能」と呼ばれるプログラムをインストールすることができ，Web ブラウザの機能を拡張することが可能である．この Web ブラウザの拡張機能 (以下，BE) は一定数のユーザに利用されている．Chromium ブログ [2] および Firefox のレポート [1] によると，Google Chrome ユーザの半数，および Firefox ユーザの 1/3 が Web ブラウザ拡張機能を利用している，と報告されている．その一方で，悪性な BE(以下，MBE) が存在している．本論文における MBE とは，ユーザ，Web サービス事業者，Web 広告事業者に不利益をもたらす拡張機能を指す．

ユーザに不利益をもたらす BE は以下の挙動を行うことが報告されている．

- ユーザの情報窃取
  - 閲覧履歴の窃取: これは BE の API を利用することで，ユーザの閲覧履歴を取得し，その閲覧履歴を MBE

[1] 電気通信大学
The University of Electro-Communications
[a)] o1830020@edu.cc.uec.ac.jp

の開発者が持つサーバに送信する挙動のことである．エンジニアによる調査 [3] では，一部の Web サイトでは認証トークンの一部に URL を利用する場合があるため，閲覧履歴の窃取は危険であると述べている．
  - 入力情報の窃取: これはユーザが Web ページに入力する情報を取得し，MBE の開発者が持つサーバに送信する挙動のことである．パスワードやログイン ID の情報が窃取される可能性があるため，危険である．
- 悪性なコンテンツへの誘導
  悪性なコンテンツへの誘導とは，MBE が (1)HTTP 通信を監視・改ざんし，フィッシングサイトやマルウェア配布サイトへ誘導する，(2)Web ページに DOM 要素を挿入もしくは，Web ページ内の DOM 要素を改ざんし，偽のアラートメッセージの挿入やマルウェアの配布を行う挙動のことである．ユーザがアクセスしたい Web サービスにアクセスできないことや悪性なコンテンツへの誘導によって，Web サービス事業者への信用が低下させる可能性があるため，この挙動を行う BE は Web サービス事業者にも不利益をもたらす．
- オンラインソーシャルネットワーキングサービス (OSN) 濫用
  OSN 濫用とは，ユーザが OSN に送信する通信，もしくは OSN 内に存在しているコンテンツを改ざんすることで，ユーザの SNS アカウントを不正利用する挙動のことである．また OSN 運営者の信用も低下する可能性があるため，OSN 濫用を行う BE は Web サービス事業者にも不利益をもたらす．

また広告の挿入・変更を行うことで，Web 広告事業者に不利益をもたらす BE が存在している．広告の挿入・変更とは Web 広告を含む DOM 要素を Web ページ内の広告の上に挿入，または Web ページに含まれている広告を別の広告に置換する挙動である．その結果，Web 広告事業者に入る収益が窃取される．また Web 広告による収益は Web サービス事業者も得られるため，MBE がその収益を窃取することで，Web サービス事業者にも不利益をもたらす．

その一方で，MBE の対策は 2 種類ある．1 つ目は BE における各種機能の利用許可である．ユーザは BE の権限を自由に消去することで，BE の機能を制限することができる．しかし．ユーザが「MBE がどの権限を利用して悪性な挙動を行うのか」や「どの BE が MBE なのか」を知ることは困難である．また Jagpal らの研究 [4] によると，良性な BE(以下，BBE) と MBE は同様な権限を利用しているため，ユーザが所持している全ての BE の権限を同一に制限すると，BBE の挙動を妨げる可能性が高い．そのため，各種機能の利用許可によって MBE の挙動を妨げることは現実的ではない．

2 つ目に，ユーザからの報告がある．ユーザが Chrome Web Store に存在している MBE を報告した際に，Chrome Web Store はその MBE を削除する場合がある．しかしながら，ユーザは MBE に関する知識がないため，BE を MBE と判別することは困難である．そのため，ユーザからの報告のみで MBE を根絶することは困難である．

これらの既存対策を補完するために，MBE 検出システムが必要である．MBE 検出システムを提案した Jagpal らの研究 [4] では，特徴として JavaScript の関数名を利用しているが．多くの MBE が利用する関数は，BBE も実行することが報告されている．そこで，本研究では “挙動の差” を特徴として利用できるのではないかと考えた．挙動とは，何らかの JavaScript コードを実行した結果である，と定義する．つまり，実行結果であり，BE のコード自体を扱うわけではない．よって良性の BE と悪性の BE の挙動にはなにかしらの差があると考え，それが悪意ある BE の検出に役に立つに違いないと考えたからである．

したがって，本研究ではこの仮説を検証するため BE の挙動情報を動的解析により収集し，収集した挙動情報を挙動の差を表すデータに変換し，このデータをランダムフォレストで学習させることによって，MBE の検出器実現を試みた．

2. 検証方法

本研究では，BBE と MBE の挙動の差により，MBE の検出が可能かを検証するため，以下に示す 3 つの手順でその検証を行った．本章では，それらの手順のうち Step2 までの処理について説明を行う．

Step 1: 動作情報取得処理
Step 2: データセット構築
Step 3: 機械学習による評価

2.1 挙動情報取得処理

本研究では，まずはじめに各 BE の挙動に関する情報 (以降，挙動情報と呼ぶ) を収集する必要がある．BE の挙動情報を収集するためには，(1)BE を Web ブラウザにインストールし，かつ (2)BE が動作する Web ページへアクセスして BE を動作させる必要がある．そこで我々は，Selenium[16]+Chrome Driver[17] を用いて，動的解析によって挙動情報を収集する情報収集システムを実現した．システムの概要図を図 1 に示す．情報収集システムは Web ページと Google Chrome のデバッガ [11] から挙動情報を収集し，保存する．

情報収集システムは BE として実装されており，(BE,URL) を入力として，対象 BE に関する 1 つの挙動情報が得られる．得られる挙動情報の概要は図 2 の通りである．図 2 中において挙動と書かれているものは以下の 5 種類ある．

挙動 A: Web ページへの遷移
挙動 B: DOM 要素の追加

表 1　挙動の種類とそのログの記録例

	挙動 A	挙動 B	挙動 C	挙動 D	挙動 E
挙動の説明	ユーザが閲覧中の Web ページを変更	DOM 要素を Web ページに挿入	DOM 要素の 属性値の変更	Web ページから HTTP 通信	JavaScript コードの実行
挙動が発生する行為	悪性コンテンツ誘導	悪性コンテンツ誘導, 広告挿入・変更, OSN 濫用	悪性コンテンツ誘導, 広告挿入・変更, OSN 濫用	ユーザの情報窃取	悪性コンテンツ誘導, 広告挿入・変更, OSN 濫用, ユーザの情報窃取
各挙動情報に含まれる値	**newURL:** 変更後の Web ページの URL **oldURL:** 変更前の Web ページの URL	**Parent:** 挿入された DOM 要素の親要素名 **Nodename:** 挿入された DOM 要素名 **Type:** DOM の変更を 監視する手法 **Target_tagname:** 挿入される DOM 要素名	**attribute:** DOM 要素の 変更された属性値名 **Old_value:** 変更前の属性値 **New_value:** 変更後の属性値 **Type:** DOM の変更を 監視する手法 **Target_tagname:** 属性値が変更された DOM 要素名	**destURL:** HTTP 通信先 URL **Method:** HTTP Method **srcURL:** HTTP 通信を発生 させた Web ページや ファイルの URL	**srcURL:** 実行した Script の 発行元 URL

挙動 C: DOM 要素の属性値の変更
挙動 D: Web ページからの HTTP 通信
挙動 E: JavaScript コードの実行
つまり図 1 で示した情報収集システムでは，これらの 5 種類の挙動を収集している．**表 1** は，この各挙動において，挙動が発生すると想定される MBE の処理，各挙動情報に含まれる値が示されている．各ログの値は文字列型で記録される．また情報収集システムは BE の挙動について情報収集を行うために，Web ページ内の全てのオブジェクトの読み込みが終了してから 10 秒待機する．

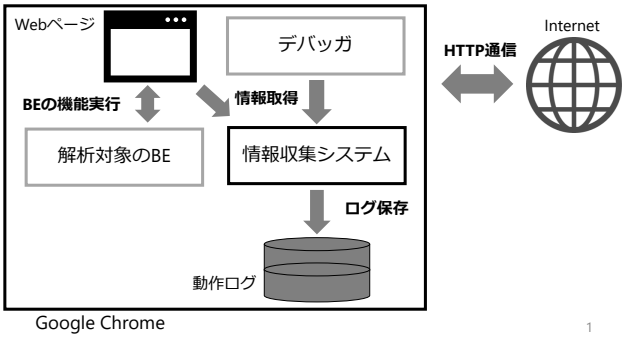


図 1　情報収集システムの概要

本研究では入力とする URL として (1) 解析対象の BE のソースコードに含まれる URL と (2)Alexa top ranking[13] 上位 10 件の Web サイトを利用した．なぜなら，解析対象 BE のソースコードに含まれている URL は，BE の機能を実行するために必要な可能性があるためである．また BE が機能を実行するために特定の DOM 要素が必要な場合があるため，これらの Web サイトを利用した．これらの Web サイトのみでは，BE を動作させる DOM 要素を得るために不十分な可能性があることは本研究の限界である．

```
Struct BE 挙動情報 {
    BE の識別子
    挙動 A のリスト
    挙動 B のリスト
    挙動 C のリスト
    挙動 D のリスト
    挙動 E のリスト
}
```

図 2　挙動情報の概要

2.2　データセット構築

前節の説明により，1 つの情報収集対象 BE について URL 数分の挙動情報が収集された．これを要素情報とし，BBE と MBE の “挙動の差” を示す情報を構築して，機械学習におけるデータセットとする．本節では，この「挙動の差を表すデータセット」の構築方法について説明する．

まず本研究における BBE と MBE の定義を以下に示す．

- BBE: MBE として報告されておらず，Chrome Web Store に存在する Chrome 拡張機能 (以下，CE と呼ぶ)
- MBE: Web 記事にて，本研究が対象とする悪性挙動が報告されている CE

なお評価実験を行うにあたり，手作業で BBE を 937 件，MBE を 9 件収集し，データセットに利用した．ここで MBE は 2016 年から 2018 年に掲載された Web 記事に悪性な挙動が記載されている CE を利用している．なお MBE の挙動のうち，OSN 濫用を行う MBE を入手できなかったため，OSN 濫用は本研究では評価できていない．

次に，以下に示す手順により 2 つの挙動集合を作成する．
Step 1: n 個の BBE に関する動作情報から特定挙動の情報

を抽出し，1 つの集合を作成する．

Step 2: Step1 で作成した集合から重複するデータを削除し，ユニークな挙動の集合とする．

Step 3: 1 個の MBE に関する動作情報から特定挙動の情報を抽出し，1 つの集合を作成する．

Step 4: Step2 の処理を Step3 で作成した集合に対して行う．

表 2 学習に使用される特徴量

特徴量	型
Web ページへの遷移 (*Con*)	float
Web ページへの遷移 (*Sub*)	float
DOM 要素の追加 (*Con*)	float
DOM 要素の追加 (*Sub*)	float
DOM 要素の属性値の変更 (*Con*)	float
DOM 要素の属性値の変更 (*Sub*)	float
Web ページからの HTTP 通信 (*Con*)	float
Web ページからの HTTP 通信 (*Sub*)	float
JavaScript コードの実行 (*Con*)	float
JavaScript コードの実行 (*Sub*)	float

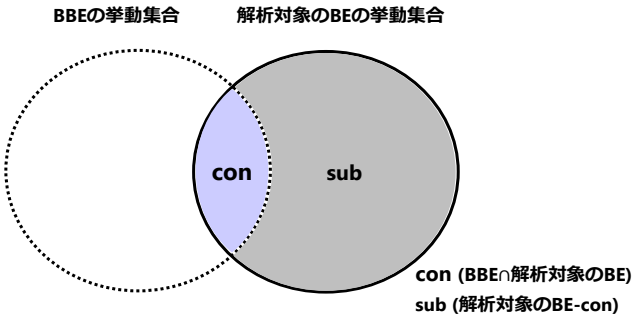


図 3 解析対象の BE と BBE の挙動集合

これにより，特定挙動に関して BBE の集合と MBE の集合ができたことになる．これらの関係を図 3 に示す．この図 3 を基に，BBE と MBE の挙動の差として以下の 2 つの情報 *Con*，*Sub* を定義する．

Con: BBE の挙動集合と解析対象 BE の挙動集合の積集合．

Sub: 解析対象 BE の挙動集合から *C*on 部分を引いた差集合．

Con を定義した理由は，この集合の要素数が大きい場合，解析対象の BE は BBE と同一の挙動を多数行っていることから，BBE である可能性が高いと判断できると考えたからである．*Sub* を定義した理由は，この集合の要素数が大きい場合，解析対象の BE は BBE が行わない挙動を多数行っていることから，MBE である可能性が高いと判断できると考えたからである．したがって，特定 BE の “挙動情報” について 2 つの値が得られる．なお前節で述べた通り挙動は 5 種類あることから，特定 BE について，10 種類の値が作成される (表 2)．これら 10 種類の値をデータセットとして使用する．なお各値の型が float である理由は，もとの値である該当領域の要素数に対して，scikit-learn の MinMaxScaler モジュール [14] を用いて正規化した値を学習に用いたためである．

3. 機械学習による評価

本章では，2 章の冒頭で示した検証方法の概要のうち Step 3 にあたる「機械学習による評価」について．以下のステップにしたがって説明する．

Step 3.1: ランダムフォレストのパラメータ決定

Step 3.2: ランダムフォレストによる特徴量の評価

3.1 ランダムフォレストのパラメータ決定

2.2 節で作成したデータセットを用いて，学習モデル「ランダムフォレスト (以降，RF と呼ぶ)」で識別器の作成を試みた．その際，必要なパラメータの設定を行なったので，それについて述べる．本節で決定するパラメータは次の 2 種類である．

- BBE の挙動集合作成 (2.2 節 Step1) に用いる BBE の数 n
- RF に利用する決定木の数 $nTree$

まず BBE の数 n を決定する．今回の検証では，$n = (100, 200, 300, 400, 500)$ の 5 条件を用いて比較した．本検証には scikit-learn[14] を用いており，RF の決定木の数 $nTree$ はデフォルト値である 10 を利用し，層化 9 分割交差検証を用いた．9 分割の理由は，データセットにおける「MBE のデータ数」が「BBE のデータ数」と比較して少なく，MBE データに依拠する結果のブレをなくす必要があると考えたためである．またデータセットにおける「MBE のデータ数」が「BBE のデータ数」と比較して少ないことで，全てのデータを BBE と判断してしまう学習モデルが作成される可能性があるため，訓練データに Under-Sampling を行なっている．なお本検証の評価指標には学習モデルの評価指標の 1 つである ROC 曲線における曲線化面積 (AUC) を利用した．なぜなら AUC を利用することにより，学習モデルが「正解ラベルに近い値を算出できるか」を評価できるからである．また n がいずれの値の場合も，訓練およびテストデータに利用する BBE の数を 437 件とした．なぜなら，$n = 500$ の際に，訓練およびテストデータとして利用可能な BBE の数は合計 437 件だからである．

結果を**表 3** に示す．RF の無作為性とデータセット内 MBE の数が少量であることにより，AUC は変動する可能性があるものの，$n = 500$ で AUC 値が最大になったので，BBE の数 n は 500 とした．

次に学習モデル RF のパラメータである決定木の数の決定を試みた．このため，パラメータごとに層化 9 分割交差検証を行った．決定木の数 ($nTree$) は $(10, 10^2, 10^3, 10^4, 10^5)$ の 5 種類の値を用いた．結果を**表 4** に示す．AUC の値

を比較すると $nTree \geq 10^2$ ではほぼ同じ値となった．一方，偽陽性，偽陰性をそれぞれの条件下で比較すると，$nTree = 10^2$ の時にそれらの値が最も良い結果となった．よってこれらの結果から，今回の評価実験では決定木の数を $nTree = 10^2$ とした．

表 3 特徴量化に使用する BBE の数による RF の AUC

	100	200	300	400	500
AUC	0.608	0.677	0.718	0.762	0.798

表 4 RF におけるパラメータチューニング

	10	10^2	10^3	10^4	10^5
偽陽性	0.339	0.303	0.330	0.337	0.330
偽陰性	0.556	0.222	0.222	0.222	0.222
AUC	0.747	0.795	0.788	0.797	0.793

これまでの議論を通じて得られた MBE 識別器の評価結果と検証条件について以下にまとめる．

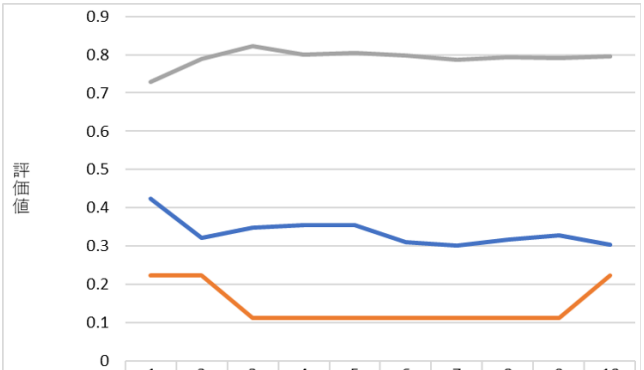
- 偽陽性 (False positive): 0.302
- 偽陰性 (False negative): 0.222
- 条件
  - 学習モデル: Random Forest (決定木数 $nTree = 10^2$)
  - 層下 9 分割交差検証
  - データセット: BBE 437 個, MBE 9 個 (訓練データは Under-Sampling 適用)
  - データの特徴量は 10 種

3.2 学習データ内特徴量の評価

前節で決定した学習モデルを用いて，学習データに含まれる 10 種類の特徴量 (**表 2**) の有用性について調査を行なった．調査は 3.1 節で得られた MBE 識別器を用いて，(1) データセットに対して層化 9 分割交差検証を行い，(2) それぞれの検証で特徴量の重要度を取得し，(3) 特徴量の重要度の平均値を計算するという手法で行った．特徴量の重要度は scikit-learn の feature_importances_から取得した．結果を**表 5** に示す．表 5 内の数値は特徴量の重要度を示しており，数値が大きいほど MBE の識別をする上でその特徴量が有用であることを示している．

この結果について 2 つの観点から述べる．まず *Con* と *Sub* の間の比較だが，*Con* の特徴量と *Sub* の特徴量の重要度をそれぞれ合計すると，*Sub* の特徴量の重要度の合計は 0.6 となり，*Con* の特徴量の重要度より 1.5 倍ほど大きな値となっている．そのため，*Sub* の特徴量の方が，*Con* の特徴量よりも MBE 識別に有益な可能性がある．

次に，各挙動毎の有用度を ($Con + Sub$) の値で比較した場合，「DOM 要素の追加」が MBE 識別において有益であることが示唆された．一方で，「JavaScript の実行」は MBE 識別において，有益ではないことが示唆された．



	1	2	3	4	5	6	7	8	9	10
偽陽性	0.424	0.321	0.348	0.355	0.355	0.309	0.302	0.316	0.328	0.303
偽陰性	0.222	0.222	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.222
AUC	0.728	0.789	0.823	0.801	0.805	0.797	0.788	0.794	0.792	0.795

図 4 特徴量の追加に伴う評価値の変化

表 5 特徴量の重要度

	Con	*Sub*	*Con+Sub*
挙動 A	0.046	0.081	0.128
挙動 B	0.125	0.334	0.460
挙動 C	0.143	0.064	0.207
挙動 D	0.090	0.073	0.163
挙動 E	0.000	0.043	0.043
値の合計	0.404	0.596	1.000

この結果をふまえ，有用度の高い特徴量だけを用いて学習を行うことで識別器の精度が向上するのではないかと我々は考えた．そこで 10 種の特徴量を表 5 の *Con*, *Sub* 値の高い順にソートし，上位から順に 1 つづつ特徴量を増やしつつ識別器の精度検証を試みた．なお特徴量の種類数以外は，3.1 節の最後に示した評価条件のもとで検証した．結果を**図 4** に示す．結果から，AUC 値は上位 2 種の特徴量を使用した場合以外，ほぼ同じ値になっていることが分かる．そのため，偽陽性，偽陰性を比較すると，結果から上位 7 種の特徴量 (挙動 A: Web ページの遷移 (*Sub*), 挙動 B: DOM 要素の追加 (*Con* & *Sub*), 挙動 C: DOM 要素の属性値変更 (*Con* & *Sub*), 挙動 D: Web ページからの HTTP 通信 (*Con* & *Sub*)) を用いた場合が最も良い結果となった．

4. 考察

4.1 先行研究との比較

Kim らは，静的解析と動的解析を組み合わせ，JavaScript コードを強制実行させることによって，Exploit Kit(以下，EK) や MBE を解析する JavaScript エンジンである “J-Force” を提案した [7]．評価実験では，Chrome Web Store 内の 12,132 件の CE から，広告挿入を行う CE 322 件，情報漏洩を行う CE 30 件を発見した．しかし，その 2 種類以外の MBE に対して評価実験はされていない．また J-Force は，MBE によって実行環境の違いを検出され，MBE の実行を困難にさせる可能性がある．

表 6　先行研究と本研究との比較

	広告挿入・変更	情報漏洩	悪性コンテンツ誘導	検出システム回避	DOM 操作の実行結果
Kim らの研究 [7]	○	○	○	×	○
Kapravelos らの研究 [8]	○	○	○	×	○
Starov らの研究 [9]	×	○	×	○	×
Chen らの研究 [6]	×	○	×	○	×
Xing らの研究 [10]	○	×	△	○	○
Jagpal らの研究 [4]	○	○	○	○	×
Wang らの研究 [5]	○	○	○	○	×
本研究	△	△	△	○	○

Kapravelos らは MBE を検知するために，MBE の動的解析システムである “Hulk” を提案した [8]．Hulk は “Honey Page” と “Fuzzer” の二つの機構から構成されている．Honey Page とは，BE が利用する JavaScript の関数を上書きすることによって，BE が DOM 要素を利用する前に，BE の挙動に必要な DOM 要素を生成し，BE の挙動を実行させる仕組みである．また Fuzzer とは 100 万の URL にアクセスしたように見せかけることで，BE に含まれている HTTP 通信に関するイベントハンドラを実行する仕組みである．Kapravelos らは公式の Chrome Web Store，非公式な BE 配布サイトから取得した BE をそれぞれ 47,940 件と 392 件を用いて，MBE が行っている活動や利用している権限を調査した．しかし，Hulk は情報漏洩，BE のアンインストール防止のどちらかの挙動を行った BE を MBE と判断するため，それ以外の挙動を行う MBE の検出を行うことはできない．また，Honey Page は JavaScript の関数を上書きするため，関数の文字数やハッシュ値が変化することが考えられる．そのため，これらの値を通じて MBE は Honey Page の存在を検知可能なため，MBE は Honey Page を回避できる．

前述の通り，MBE は J-Force[7] と Hulk[8] を回避可能である．その原因は，JS エンジンや Honey page が実環境と異なることに起因する．一方で，本研究は JavaScript エンジンに手を加えたり，HoneyPage を必要とする情報取得方法を利用していない．したがって，これらの研究と比較して実環境との差異は少ないと言える．

Starov らは BE によるユーザの情報窃取を防ぐために，“Browsing fog” を提案した [9]．Browsing fog は人気のある Web ページ 10 件とそれらの Web ページにリンクされている Web ページを模した静的 Web ページから，Honey Page[8] と mitmproxy[15] を利用することで，BE による通信を取得する．Browsing fog を利用して人気のある BE を 9,839 件調査した結果，655 件が情報漏洩をしていた．

Starov らの研究では DOM 挿入や属性値の変更結果を利用していない．本研究では 2 章，3 章で述べた通り，DOM 挿入や属性値の変更結果を情報として収集し，ユーザの情報窃取を行う MBE の検出を試みた．

Chen らは情報漏洩を発生させる BE の検知を行うシステムである “Mystique” を提案し，拡張機能のストアに存在していた CE（178,893 件）と Opera 拡張機能（2,790 件）に対して評価した [6]．静的解析と動的解析を組み合わせることによって，CE に対して 89.80%の精度で BE が情報漏洩をしているかを検出可能である．Mystique では BE の挙動実行のために，Honey Page を利用しているため，Kapravelos ら [8] と同様の限界が存在している．

Xing らは，広告の挿入を行う MBE を検出するために，Expector を提案・評価した [10]．Expector が広告を検出する手順としては，まず CE を Web ブラウザにインストールした場合としていない場合のそれぞれにおいて，複数回 Web ページを読み込んだ際の DOM を記録する．その後，DOM 要素が所持している URL にアクセスし，リダイレクトを追跡する．そして．もし DOM 要素が CE を Web ブラウザにインストールした場合のみ挿入されており，かつ複数のサイトにリダイレクトされ，最終的にコンテンツに到達する場合，その DOM 要素を Web ブラウザ拡張機能が挿入した広告と判別する．Expector は Selenium[16] と Chrome Driver[17], mitmproxy[15] を組み合わせることによって実装されている．CE 18,000 件を調査した結果，広告挿入を行っていた CE は 292 件だった．また手動検査によってこの結果を検証すると，広告判定における偽陽性は 3.7%，偽陰性は 3.0%だった．

Xing らの研究 [10] では，DOM 操作の比較を行う点と動的解析に使用するシステムが本研究と類似している．しかし，本研究では Xing らの研究 [10] とは異なり，広告挿入・変更を行う以外の挙動を行う MBE も検出を試みている．

Jagpal らの研究 [4] は分類器およびルールベースの検知機を作成し，2012 年から 2015 年までの MBE を検知を試みた．分類器では静的解析と動的解析，開発者の情報を特徴量として使用した．まず静的解析では解析対象となる BE のソースコードやディレクトリ構造を解析した．次に動的解析では，Windows マシン上で，Windows や Google Chrome の設定や環境（例:Windows の設定など）の変更をキャプチャするシステムモニタとすべての DOM イベント，Chrome API 呼び出しに介入してログを記録するブラ

ウザ内アクティビティーロガー [12] を利用して解析を行った．また Kapravelos らの研究 [8] で利用された URL と人気のあるニュース，ビデオ，ショッピング，銀行サイトに対してクローリングすることで，動的解析システムと解析対象の BE を動作させ，情報を取得した．そして，Web Store にある BE の開発者の情報 (例: 最後にログインした場所，開発者のメールアドレス) を利用している．最後にこれらの特徴量を L1 正則化によって選別し，ロジスティック回帰を用いて識別器を実現した．ルールベースの検知機では OSN 濫用，広告挿入，情報漏洩，ユーザの動作取得の 4 つに対して，それぞれルールを作成することで，検出を行った．このロジスティック回帰を用いた検知機とルールベースの検知機を組み合わせた “WebEval” を 99818 件の BE(BBE:90,295 件,MBE:9,523 件) で評価したところ，精度は 73.7%であり，再現率は 93.3%だった．

Wang らの研究 [5] では「BBE だがバグを含む BE」と MBE を検出するために，静的解析と動的解析を行って，24 種類の特徴量の集合を取得し，これらの特徴量を利用して学習を行うことで実現した識別器を検証した研究である．Wang らの研究 [5] では Jagpal らの研究 [4] と同一のブラウザ内アクティビティーロガー [12] を用いて動的解析を行っている．BBE を 3187 件，良性だがバグを含んでいる拡張機能と MBE を合わせて 1490 件収集し，この識別器を検証した結果，精度は 95.18%，偽陽性は 3.66%だった．

Jagpal らの研究 [4] と Wang らの研究 [5] では，動的解析にブラウザ内アクティビティーロガー [12] を用いている．しかし，このブラウザ内アクティビティーロガー [12] では CE が行っている JavaScript の組み込み関数名や Chrome API 名が取得できるものの，それらの実行結果については監視していない．Jagpal らの研究 [4] より，これらの値は MBE の識別に有益なものの単一の情報では識別できないことが示されている．一方で，本研究ではそれらの関数の動作を用いて特徴量化をしている．BE の挙動を監視することによって，それらの関数名よりも情報量が増加するため，BE の挙動の方が，MBE の識別に有益だと考える．

表 6 に本節で行った先行研究と本研究との比較をまとめてある．表 6 内には△が四つあるが，そのうち Xing らの研究 [10] では広告を用いた一部の悪性コンテンツへの誘導を検出できるため，△を記載した．また，本研究の広告挿入・変更，ユーザの情報漏洩，悪性なコンテンツへの誘導といった MBE 検出への有用性は暫定的な結果のみであるため，△を記載した．また表 6 内の検出システム回避は，検出可能である J-Force[7] と Hulk[8] に×を記載している．

表 6 より，本研究は広告挿入・変更，ユーザの情報漏洩，悪性なコンテンツへの誘導といった挙動を行う MBE を検出可能であり，かつ MBE による DOM 操作の実行結果を監視することが可能であることがわかる．またこれらの事柄を本研究と同様に行うことが可能な J-Force[7], Hulk[8] と本研究を比較すると，本研究で実装した情報収集システムは実環境との差異は少ないと考える．

4.2 特徴量の重要度

3.1 節より，偽陰性と偽陽性が最も良い学習モデルには 10 種類の特徴量のうち，7 種類が利用されていた．本節ではこの学習モデルに使用されなかった 3 種類の特徴量 (「挙動 E: JavaScript の実行 (*Con* & *Sub*)」と「挙動 A:Web ページへの遷移 (*Con*)」) について述べる．

まず挙動 E(*Con*) について，BBE は CDN といった外部サーバから JavaScript のライブラリを利用し，MBE はそのライブラリを改ざんして悪性な挙動を埋め込む可能性があるという仮説を立てた．しかし，ほとんどの BE は BE 内に JavaScript のライブラリを所持しており，拡張機能 ID の違いからそれらのライブラリは同一のスクリプトとして監視できなかった．実際に 3.1 節で述べたデータセットによると，挙動 E(*Con*) の中央値は BBE，MBE ともに 1.0 であった．この値は情報取得システムの実行を示すログであるため，解析対象の BE によるログではない．また挙動 E(*Sub*) においても，ほとんどの BE が実行した JavaScript は同一とみられなかったため，RF が BE と MBE の違いを発見することは困難であったと考えられる．

また「挙動 A(*Con*)」は，本検証において Alexa Top Site[13] を利用したことにより，挙動 A(*Con*) の多くがこれらの Web サイトであったため，MBE 識別に利用できなかった．実際に，3.1 節で述べたデータセットによると，挙動 A(*Con*) の中央値は BBE は 12，MBE は 11 であった．

4.3 本研究の限界

本研究では BE のうち，CE に対象を絞っている．情報取得システムは Google Chrome のデバッガ [11] を利用しているため，CE 以外の BE に対しては適応できない．他の Web ブラウザへの対応は今後の課題である．

また情報収集システムは Web ページが行った DOM 操作と BE が行った DOM 操作を区別できていない．しかしながら，BBE によってアクセスされた Web サイトの DOM 操作は *Con* として演算されるため，特徴量化に利用する BBE の数を増加した場合，Web サイトによる DOM 操作の影響は軽減される可能性がある．なぜなら，BBE と同一の Web サイトに多くアクセスしている BE は BBE の可能性があり，BBE と異なる Web サイトに多くアクセスしている BE は MBE の可能性があると考えるからである．

先行研究と比較して，本研究ではデータセットが小規模であり，かつ BBE と MBE のデータ数にも大きな差がある．特に MBE は 9 件しか取得することができなかった．なぜなら本研究で用いた MBE は Web 記事として報告されていた MBE のみを利用したからである．このデータセットを利用して行った今回の評価結果は MBE の網羅性が不

十分であることと学習モデルの過学習が発生している可能性がある．そのため，本研究で用いた特徴量の有用性検証はは暫定値であると言わざるをえない．また本研究ではランダムフォレストのみで評価を行ったが，今回提案した特徴量に最適な学習モデルを検討する必要がある．今後はより大規模なデータセットを用意し，複数の学習モデルで評価することによって，本研究で評価した特徴量の有効性検証と最適な学習モデルの提案を行う予定である．

本研究では BE の動的解析を行っているため，MBE のコードカバレッジは問題である．そのため，将来的には静的解析を利用して，多様な特徴量を使用することにより，動的解析のみに依存しない学習モデルの提案を行うことで，この問題の解決を試みる．また大規模な数の Web サイトに対して解析対象の BE と情報収集システムをインストールした Google Chrome でアクセスすることで，BE が機能実行に必要となる DOM 要素を含んでいる Web サイトにアクセスし．コードカバレッジを向上させる予定である．

5. おわりに

悪性な Web ブラウザ拡張機能はユーザや Web サービス運営者，Web 広告事業者に対する脅威である．既存の対策では，良性な Web ブラウザ拡張機能の権限を制限することとユーザによる MBE の報告が挙げられるが，いずれも悪性な Web ブラウザ拡張機能をなくすことはできない．そのため本研究では，悪性な Web ブラウザ拡張機能と良性な Web ブラウザ拡張機能の”挙動の差”を特徴量とするデータを用いて悪性な Web ブラウザ拡張機能の検出を試みた．良性な Web ブラウザ拡張機能 937 件と悪性な Web ブラウザ拡張機能 9 件で学習データを作成し，ランダムフォレストを用いて，層下 9 分割交差検証法で評価した．結果として，決定木の数を 100 とした際のランダムフォレストにおいて，偽陽性は 30.2%，偽陰性は 22.2%だった．また本研究では，個々の特徴量の有用性評価から，「追加された DOM 要素」に着目することは MBE 識別に有益であることが示唆された．今後はより規模の大きなデータセットを用意し，今回の検証方法で再び検証を行う予定である..

謝辞　データセットに利用した Web ブラウザ拡張機能の収集に協力して頂いた所属研究室諸氏，および本研究において Web ブラウザ拡張機能に含まれた URL を取得するために静的解析ツールを利用させていただいた「Team UN 頼み」の皆様に心より御礼申し上げます．

参考文献

[1] Mozilla: Firefox Public Data Report, Firefox Public Data Report(オンライン), 入手先〈https://data.firefox.com/dashboard/usage-behavior〉

[2] Google: Trustworthy Chrome Extensions, by default, Chromium Blog（オンライン），入手先〈https://blog.chromium.org/2018/10/trustworthy-chrome-extensions-by-default.html〉

[3] Robert Heaton: ”Stylish” browser extension steals all your internet history — Robert Heaton，Robert Heaton（オンライン），入手先〈https://robertheaton.com/2018/07/02/stylish-browser-extension-steals-your-internet-history/〉

[4] Jagpal N., Dingle E., Gravel J.P., et al.: Trends and Lessons from Three Years Fighting Malicious Extensions，Proc. *24th USENIX Security Symposium*，(2015).

[5] Wang Y., Cai W., Lyu P., et al.: A Combined Static and Dynamic Analysis Approach to Detect Malicious Browser Extensions，*Security and Communication Networks*，vol. 2018, Article ID 7087239, 16 pages, 2018. https://doi.org/10.1155/2018/7087239.

[6] Chen Q. and Kapravelos A. : Mystique: Uncovering Information Leakage from Browser Extensions，Proc. *the 2018 ACM SIGSAC Conference on Computer and Communications Security*，(2018).

[7] Kim K., Kim I.L., Kim C.H., et al.: J-Force: Forced Execution on JavaScript，Proc. *the 26th International Conference on World Wide Web*，(2017).

[8] Kapravelos A., Grier C., Chachra N., et al.: Hulk: Eliciting Malicious Behavior in Browser Extensions，Proc. *the 23rd USENIX Security Symposium*，(2014).

[9] Starov O. and Nikiforakis N.: Extended Tracking Powers: Measuring the Privacy Diffusion Enabled by Browser Extensions，Proc. *the 26th International Conference on World Wide Web*，(2017).

[10] Xing X., Meng W., Lee B., et al.: Understanding Malvertising Through Ad-Injecting Browser Extensions，Proc. *the 24th International Conference on World Wide Web*，(2015).

[11] Google:Chrome DevTools Protocol Viewer, Github Pages（オンライン），入手先〈https://chromedevtools.github.io/devtools-protocol/〉

[12] Google: Chrome Apps & Extensions Developer Tool - Chrome Web Store，Chrome Web Store（オンライン），入手先〈https://chrome.google.com/webstore/detail/chrome-apps-extensions-de/ohmmkhmmmpcnpikjeljgnaoabkaalbgc〉

[13] Alexa: Alexa - Top sites，Alexa（オンライン），入手先〈https://www.alexa.com/topsites〉

[14] Pedregosa, F., Varoquaux, G., Gramfort, A., et al.: Machine Learning in Python，*Journal of Machine Learning Research*，vol. 12,pp. 2825-2830, (2011).

[15] mitmproxy: mitmproxy - an interactive HTTPS proxy，mitmproxy(オンライン), 入手先〈https://mitmproxy.org/〉

[16] Seleium: Selenium - Web Browser Automation, Selenium(オンライン), 入手先〈https://www.seleniumhq.org/〉

[17] Chromium project: ChromeDriver - WebDriver for Chrome, ChromeDriver(オンライン), 入手先〈https://www.seleniumhq.org/〉

[18] Brian Cherne: briancherne/jquery-hoverIntent: hoverIntent jQuery Plug-in, GitHub(オンライン), 入手先〈https://github.com/briancherne/jquery-hoverIntent〉