Canvas Fingerprintingを用いたWebトラッキングの検証と実態調査

芳賀 夢久† 高田 雄太†‡ 秋山 満昭‡ 森 達哉† 後藤 滋樹†

† 早稲田大学 基幹理工学研究科
169-8555 東京都新宿区大久保 3-4-1
{yumehisa.haga,mori}@nsl.cs.waseda.ac.jp, goto@goto.info.waseda.ac.jp

‡NTT セキュアプラットフォーム研究所
180-8585 東京都武蔵野市緑町 3-9-11
{takata.yuta,akiyama.mitsuaki}@lab.ntt.co.jp

あらまし HTML5 では Canvas 要素を用いてブラウザ上に図を描画することができるが，生成される画像はユーザーの OS やブラウザによりピクセル単位の違いが生じる．この差を利用してブラウザを一意に特定する技術を Canvas Fingerprinting と言う．Canvas Fingerprinting を用いてユーザーの行動を追跡する Web サイトは増加傾向にある．本研究では良性な Web サイト約 100 万件とブラックリストに登録された悪性な Web サイト約 300 万件を対象とした大規模な Web トラッキングの実態調査を行った．調査の結果，良性・悪性ともに多くの Web サイトで Canvas Fingerprint を含む複数の Fingerprint を同時に採取するスクリプトが発見された．さらに様々なクライアント環境における Canvas Fingerprint を収集・分析した結果，従来の Canvas Fingerprinting の検知手法は容易に回避できることを明らかにした．

A Large-scale Study of the Canvas Fingerprinting and Its Implications

Yumehisa Haga† Yuta Takata†‡ Mitsuaki Akiyama‡ Tatsuya Mori†
Shigeki Goto†

†School of Fundamental Science and Engineering, Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo 169-8555
{yumehisa.haga,mori}@nsl.cs.waseda.ac.jp, goto@goto.info.waseda.ac.jp

‡NTT Secure Platform Laboratories
3-9-11 Midoricho, Musashino-city, Tokyo 180-8585
{takata.yuta,akiyama.mitsuaki}@lab.ntt.co.jp

Abstract In HTML5, we can draw figures on a web browser by Canvas. The drawn image varies in pixels depending on users' Operating Systems and browsers. In recent years, many websites try to identify users' browsers for targeting advertisements by the Canvas fingerprint. It is also concerned that web tracking may be used for malicious activities. In this paper, we investigate popular and malicious Web cites to clarify whether they perform tracking, and founnd many Web sites have scripts that collecting multiple Fingerprints including Canvas Fingerprint. We also implement a web service for collecting Canvas fingerprint to understand the features of Canvas images, and aim to detect or block Web tracking. After we analyzed the collected data, it was revealed that the conventional Canvas Fingerprinting detection method can be avoided easily.

1 はじめに

背景: 近年，Web トラッキング技術を用いたターゲティング広告を行う Web サイトが増加傾向にある．Web トラッキングとは，特定ユーザーのサイト内での動きを追跡・分析する技術である [1–7]．これにより，ユーザーの趣味嗜好にあった広告を提供したり，マーケティングに活用することができる．一方で，

こういったトラッキング行為がユーザーのプライバシーを侵害したり，フィッシングなどに悪用されたりすることが懸念されている．2013 年には，米国において Facebook が利用者のネット履歴を不正に追跡したとして，和解金 950 万ドルを支払うといった訴訟が起こっている [8]．

ユーザーのブラウザを一意に識別するデータとして，HTTP クッキーが広く利用されている．しかし，HTTP クッキーはユーザーの操作で簡単に削除，ブロックができ，さらに Same-Origin Policy によりドメインをまたいでユーザーを追跡することはできない．この制限を克服するために，Web 管理者は Web Browser Fingerprint（以下，**WBF** と省略）と呼ばれる HTTP クッキー以外の情報を用いてユーザーを識別するようになった．WBF とは，ユーザーのブラウザの種類，画像解像度，プラグインの名前，インストール済みフォントなどの特徴点を組み合わせたものであり，Eckersley [1] によると，Flash か Java 仮想マシンのいずれかが有効になっているブラウザにおいて，これらの情報を元に 94.2%の確率でユーザーを特定可能とある．

また 2012 年には，新たな Fingerprint として，Canvas Fingerprint（以下，**CF** と省略）が Mowery ら [2] によって提案された．Canvas とは HTML5 に追加された要素であり，ブラウザ上に様々な図形や文字を描画することができる．その描画結果をピクセル単位で見ると，OS やブラウザ，インストールされているフォントの組み合わせによって微妙に異なっている．さらに，ブラウザ上に 3D 映像を描画する WebGL を使用することで，ユーザの PC の GPU も識別することができる．これを Fingerprint として利用し，ユーザーの環境を特定することができる．

CF には以下に示す 3 つの特徴がある．1) HTTP クッキーと異なり，ドメインをまたいだサイトでも同じ値を採取可能．2) User-Agent と異なり，ブラウザの機能で値を書き換えることが不可能．3) 通常利用とトラッキング目的での利用の区別が困難．Acar ら [3] の 2014 年の調査によると，Alexa [9] の上位 10 万サイトのうち，約 5.5%のサイトが Canvas を用いたトラッキングを行うスクリプトを含んでいることが明らかになった．今後も CF を用いた Web トラッキングの増加が予想される．

狙いとアプローチ: 本研究では，WBF の中でも提唱されたのが比較的新しく，対策や実態調査があまり進んでいない CF に着目する．本研究の目的は大きく 2 つある．ひとつはインターネット上の Web サイトを調査し，Canvas がどのような使われ方をしているのかを分析することである．本研究では Alexa の上位 100 万サイトに加え，Blacklist に登録されている悪性サイト約 300 万サイトをクロールした．もう一つの目的は，CF の特性を深く理解し，トラッキングの検知，防御に活かすことである．これを達成するために，本研究では CF を収集する Web サイトを作成し，さまざまな図形を描画した結果を収集，分析する．

貢献: 本研究の主要な貢献を以下に示す．

- Web サイトを広範囲にクロールし，Canvas を用いているサイトの最新の実態を明らかにした．Alexa の上位 100 万サイトでは，CF を採取していると考えられるサイトが少なくとも約 2 万サイト存在し，そのうちの 8 割以上は CF と同時に他の WBF も採取していることがわかった．
- これまで明らかになっていなかった CF の特性を発見し，既存研究で提案された CF 採取の検知手法は容易に回避できることを示した．
- CF の特性を理解する過程で，一意性がより高い新たなフィンガープリントとして CSS Fingerprint を発見した．

本研究で作成した CF 収集サイトは `http://canvas.nsl.cs.waseda.ac.jp/` で公開しており，ユーザーは自分のブラウザの Fingerprint の分析結果を閲覧することができる．また，本研究で発見した CSS Fingerprint はこれまでの CF よりもユーザー識別により寄与する特徴点であることが分かり，今後対策を急ぐべきである．

本論文の構成は以下のとおりである．はじめに 2 章で関連研究を示す． 3 章では Web サイトの実態調査の結果を述べ， 4 章で CF の拡張性について述べる． 5 章で本研究の制限と今後の展望について述べ，最後に 6 章にて本論文のまとめを述べる．

2 関連研究

本章では，Canvas を含む WBF に関連するいくつかの研究を述べる．

Eckersley [1] らは Web トラッキングにおける WBF の有効性を初めて実証した．彼らは Fingerprint として User-Agent や HTTP Accept Header，画面解像度，タイムゾーン，ブラウザのプラグイン，インストール済みフォントなどを利用し，ユーザーを 94.2%の割合で一意に識別することができることを示した．その後，多くの関連研究が行われ，パフォーマンス測定 [4]，JavaScript エンジン [5]，レンダリングエンジン [6] などがユーザーを識別する Fingerprint として利用できることが明らかとなった．

Canvas の画像データが Fingerprint になりうるということは，2012 年に Mowery ら [2] の研究によって明らかになった．JavaScript の記述によって Canvas

表 1: Canvas を使用している Web サイト

URL List	Crawled	Sites using Canvas	Rate %
Alexa	1,000,000	250,004	25.0
Blacklist	3,011,619	228,210	7.6

上にテキストを描画したところ，OS やブラウザの種類によって画像がピクセル単位で異なることが判明した．また，指定したフォントがその端末にインストールされていない場合，代替のフォントが使われることがわかった．本研究では Canvas の描画にテキスト以外の図形を用いた場合の検証も行う．

Acar ら [3] は 2014 年に CF を採取しているサイトの実態調査を行った．その結果，Alexa 上位 10 万サイトのうち，5,542 サイトが CF 採取を行うスクリプトを含んでいることがわかった．またそのうちの 5,282 サイトのスクリプトは AddThis [10] というサードパーティのスクリプトであった．Acer らは Canvas を用いたトラッキングの独自の検知手法を提案している．本研究では，この検知条件の妥当性を評価するとともに，より広範囲の Web サイトのクロールを行う．

3 実態調査

本研究ではインターネット上の Web サイトで Canvas がどのような使われ方をしているのかを調査するために，大規模なクロールを行った．本章ではクロールの手法と，収集したデータの分析結果を示す．

3.1 Web サイトのクロール

本研究でクロールの対象とするのはアクセス数の多い人気サイト (以下，良性サイト) と，BlackList に登録されている悪性サイトである．良性サイトのリストとして，Alexa [9] のランキング上位 100 万サイトのリストを採用した．また悪性サイトのリストとして，URLBlacklist.com [11] のリストを採用した．この Blackilist は “ads”, “adult”, “phishing” など，計 111 のカテゴリに分けられているが，本研究では “whitelist”除くすべてのカテゴリを調査対象とし，計 3,011,619 サイトをクロールする．

それぞれのサイトにおいて，トップページにアクセスし，HTML ファイルおよび script タグの src 属性でリンクされているスクリプトファイルをダウンロードし，それらを静的解析する．ダウンロードには wget を用いる．クロール期間は Alexa が 2015 年 7 月 21 日から 2015 年 7 月 23 日，Blacklist が 2015 年 7 月 24 日から 2015 年 8 月 1 日である．

3.2 Canvas の利用状況

まずはじめに，Alexa サイトと Blacklist サイトそれぞれについて，Canvas を使っているサイトがどれくらいあるのかを調査した．Canvas を利用しているか否かは JapaScript コードの中で getContext メソッドを用いているかどうかで判断することができる．本研究では，このメソッドの文字列が JavaScript コードの中に含まれているかを調査する．Alexa サイトおよび Blacklist サイトそれぞれについて，クロールしたサイトの数と，Canvas を利用しているサイトの数を表に示す．クロールしたサイトの中には，サーバの移動などの理由によりレスポンスを返さなかったものも含まれる．

調査の結果，Alexa においては 25.0%，Blacklist においては 7.6%のサイトにおいて Canvas が使用されていることがわかった．以降の分析は，これらの Canvas が使用されてるサイトを対象とする．

次に，Canvas を使用しているサイトにおいて，どのようなメソッドが用いられているのかを調査する．Canvas の Reference [12] を参照して Canvas に関連するあらゆるメソッドを検索し，使用頻度が多かったものを図 1 にまとめる．Alexa と Blacklist のグラフを比較すると，それぞれ異なる分布を示している．Alexa では，toDataURL メソッドや getImageData メソッドなど，Canvas の画像データを抽出するメソッドが多く，約 14 万サイトで使用されている．これらのメソッドは，Canvas に図形を描画する際に通常使われることはなく，逆に Fingerprint を採取するときに必要なメソッドである．Alexa になぜこのようなサイトが多いのか調査したところ，WordPress [13] を利用しているサイトにおいて，絵文字を表示するために自動的に挿入される JavaScript コードの中に toDataURL メソッドや getImageData メソッドが含

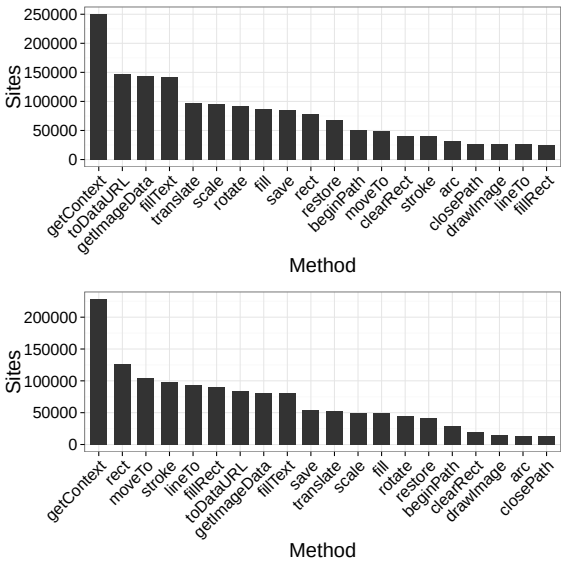

図 1: Canvas メソッドの使用頻度:Alexa(上)，Blacklist(下)．

まれることがわかった．このようなサイトは Alexa において 132,225 サイト存在することがわかった．これらのサイトはトラッキング目的で Canvas を使用している可能性は低い．一方 Blacklist では，rect メソッド，moveTo メソッドなど，Canvas に図形を描画するのに必要なメソッドが多く用いられている．なお，Blacklist において前述した WordPress のスクリプトを含むサイトは，66,755 サイト存在した．

次に，script タグの src 属性でリンクされている JavaScript ファイルのうち，外部のドメインのもの (以下，サードパーティスクリプト) について分析する．Alexa，Blacklist それぞれについて，リンクされている数が多かったサードパーティスクリプトの一覧を表 2，表 3 に示す．ただし，これらのスクリプトはすべて toDataURL メソッドまたは getImageData メソッドを用いているものであり，Fingerprint を採取している可能性が高いと推測できるものである．Alexa の中で最も多かった static.boo-box.com の embed.js というファイルを分析したところ，toDataURL メソッドを用いて Canvas の画像データを外部に送信しており，さらに User-Agent，画面解像度，CPU コア数，プラグインの種類などの情報も送信していたため，トラッキングを行っている可能性が高いと推察できる．このスクリプトでは，Canvas 上に *“Cwm fjordbank glyphs vext quiz”* という文字列を描画しているが，これはアルファベットのすべて文字を含むパングラムという言葉である．パングラムは Canvas 画像のエントロピーを高める方法として知られ，Acar ら [3] が報告している．また，tags.bkrtx.com の bk-coretag.js というスクリプトも，Canvas の画像データと共に User-Agent，MIME Type，プラグインの種類，タイムゾーンなどの情報を外部に送信しており，トラッキングを行っている可能性が高い．このスクリプトにおいても，Canvas では *“Mr. Jock, TV quiz Ph-D, bags few lynx!”* というパングラムが用いられている．Blacklist で多かった static.tumblr.com の lookbook.min.js というスクリプトも，同様に Fingerprint を採取する処理を行っていた．

ユーザーの操作では容易に削除できない Supercookie [14] を収集する JavaScript API として知られている Evercookie [15] のスクリプトファイル (ファイル名が “evercookie.jp”) は，Alexa において 77 サイト，Blacklist において 27 サイト存在した．また，ファイル名に “fingerprint”が含まれるスクリプトは Alexa において 128 サイト，Blacklist において 116 サイト存在した．第 2 章で述べたように，Acar ら [3] の報告では Alexa10 万サイトのうち，AddThis の core130.js というスクリプトを含むサイトが 5,282 サイト存在したとあったが，今回の調査では Alexa100 万サイトのうち，そのようなサイトは 2 サイトしか存在せず，類似するスクリプトもほとんど無かった．AddThis は Acar ら [3] の発表などを受けて，CF 採取を取りやめたことも考えられる．

表 2: Canvas を使用しているサードパーティスクリプト (Alexa)

Domain	File Name	Sites
static.boo-box.com	embed.js	308
tags.bkrtx.com	bk-coretag.js	170
s.atemda.com	Admeta.js	71
cdn.jsdelivr.net	wp-slimstat.js	67
www.bkrtx.com	bk-static.js	37
assets.cobaltnitra.com	bundle.js	34
cdn3.bigcommerce.com	foundation.min.js	30
images.wambacdn.net	core.js	28
static.tumblr.com	lookbook.min.js	28
c1.wikicdn.com	vkz515xz-editorc.min.js	24

表 3: Canvas を使用しているサードパーティスクリプト (Blacklist)

Domain	File Name	Sites
static.tumblr.com	lookbook.min.js	215
cdn-static.acp.adultcentro.com	compil.gz.js	147
assets.neo.registeredsite.com	easeljs.min.js	62
assets.neo.registeredsite.com	galleria.min.js	62
assets.neo.registeredsite.com	html2canvas.js	62
opengraphprotocol.org	galleria.js	55
tags.bkrtx.com	bk-coretag.js	53
cdn.jsdelivr.net	wp-slimstat.js	52
smut.com	main.js	49
images.wambacdn.net	core.js	39

3.3 他の Fingerprinting との共起

CF は，それ単体でユーザーを完璧識別することは困難であり，前節でも述べたように他の Fingerprint と組み合わせる必要がある．本節では，Canvas を使っているサイトのうち，他の WBF を採取しているサイトについて分析する．

まず，JavaScript で採取可能な WBF について分析を行う．JavaScript では User-Agent やプラグインの種類，画面解像度などの様々なプロパティにアクセスできる．その中で Acer ら [16] や，磯ら [7] が指摘する WBF になりうる情報を採取しているサイトを調査する．JavaScript で採取可能な WBF の種類と，それらを採取しているスクリプトを含むサイトの数を表 4 に示す．ここでは，CF を採取していると判定する条件を,「toDataURL メソッドまたは getImageData メソッドを使用している」とする．この条件に当てはまる良性サイトは 152,871 サイト，悪性サイトは 76,235 サイトあるが，このうち，3.2 節で述べた WordPress のスクリプトを含むサイトを除くと，良性サイトは 20,646 サイト，悪性サイトは 9,480 サイトとなる．Alexa，Blacklist 共に，User-Agent，プラグインの種類，画面のサイズなどの情報を採取しているサイトが多い．また，表 4 のプロパティをそれぞれ 1 つとカウントし，同一のスクリプトファイル内において同時に採取しているプロパ

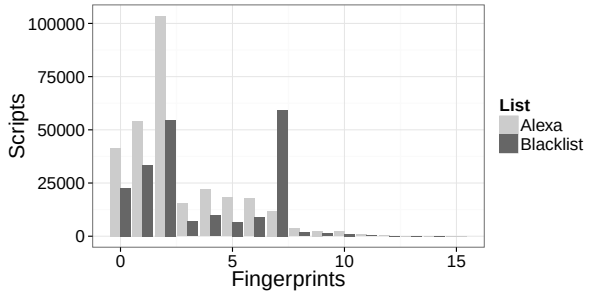

図 2: 同一スクリプトで収集される Fingerprint の数

表 5: Web ビーコンの例 (Alexa)

Element	Source	Sites
img	www.paypalobjects.com/*/scr/pixel.gif	1646
	www.googleadservices.com/pagead/conversion/*	1631
	pixel.quantserve.com/pixel/p-52ePUfP6_NxQ_.gif	59
iframe	fls.doubleclick.net/activityi*	1643
	s.thebrighttag.com/iframe?c=*	421
	google-analityks.wha.la/?0e2G	10

ティの数を集計したものを図 2 に示す．Alexa においては，4〜6 個の WBF を採取しているスクリプトが 1 万以上存在し，これらのサイトはトラッキングを行っている可能性が多い．Blacklist においては，7 個の WBF を同時に採取しているスクリプトが非常に多い．CF を採取しているスクリプトのうち，他の WBF も同時に採取しているスクリプトの割合は，Alexa では 83.9%，Blacklist では 84.5%であった．

次に Web ビーコンについて分析する．Web ビーコンとは，Web ページに埋め込まれた 1×1 pixel の画像のことで，ユーザーのアクセス動向などの情報を収集するために利用される．この Web ビーコンを見つけるために，HTML ソースの img 要素と iframe 要素を読み取り，height と width が共に 1 に指定されているものを抽出する．抽出された Web ビーコンのうち，多くのサイトに埋め込まれていたものを表 5，表 6 に示す．Alexa，Blacklist 共に Google AdSense や Paypal などによる Web ビーコンが多く存在した．CF を採取しているサイトのうち，Web ビーコンを含むサイトの割合は，Alexa では 14.9%，Blacklist では 9.6%であった．

表 6: Web ビーコンの例 (Black)

Element	Source	Sites
img	www.googleadservices.com/pagead/conversion/*	2749
	adserver.adtechus.com/adserv/3.0	2422
	www.paypalobjects.com/*/scr/pixel.gif	867
iframe	s.thebrighttag.com/iframe?c=*	147
	flex.atdmt.com/mstag/tag/*	121
	adwords-conversion-iframe.php	100

4 Canvas Fingerprinting の拡張性

本研究では Canvas 画像の特性を深く理解するために CF を収集する Web サイトを作成し，Web トラッキングの検知，防御に活かすことを目指す．本章では，作成した Web サイトの概要とデータの分析結果を述べ，さらに既存研究の CF 採取の検知手法について考察を行う．

4.1 Canvas Fingerprint 収集サイト

本研究では CF を収集するサイト (http://canvas.nsl.cs.waseda.ac.jp/canvas/) を作成し，収集データの分析を行う．システムの概要を図 3 に示す．

データ収集の手順は以下のとおりである．

1. Canvas に描画する 2 次元の図形のスタイルを JavaScript で記述し，ブラウザ上に表示させる．
2. toDataURL メソッドを用いて，Canvas 上の図形を画像データとして出力する．
3. 画像データを MD5 ハッシュ化し，それを Fingerprint として Web サーバの DB に送信する．

ユーザーがトップページのプライバシーポリシーに同意し次のページへ遷移すると，Fingerprint 採取が開始される．さらにユーザーは Canvas 画像のハッシュ値の User-Agent 毎のエントロピーなど，各種統計情報を閲覧することができる．ある Canvas 画像のハッシュ値 X の値が x_i である確率を $P(X = x_i)$ とすると，X のエントロピー $H(X)$ は以下の式で表される．

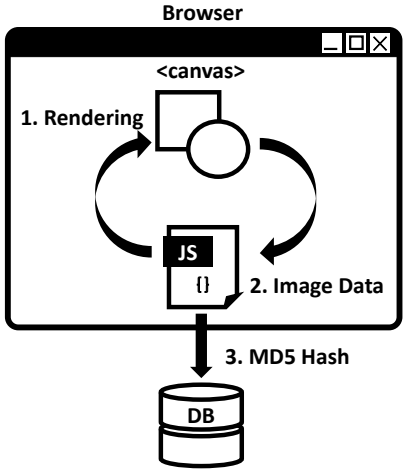

図 3: Canvas Fingerprint 収集システムの概要

表 4: JavaScript で採取可能な Web Browser Fingerprint

Property or Method	Description	Alexa	Blacklist
toDataURL(), getImageData()	Canvas image data	152871	76235
navigator.appCodeName	Code name of the browser	980	570
navigator.appName	Name of the browser	4923	11657
navigator.appVersion	Version of the brower	4453	11726
navigator.userAgent	User-Agent	34293	99392
navigator.mimeTypes	MIME types	4397	60804
navigator.plugins	Installed plugins	131546	82320
navigator.language	Language of the brower	47016	69872
navigator.platfoorm	Platfrom of the browser	47899	14520
screen.height, screen.width	Size of screen	175266	141040
screen.availHeight, screen.availWidth	Size of screen (excluding the Windows taskbar)	3334	1644
screen.colorDepth	Bit depth of the color palette	6683	2360
screen.pixelDepth	Color resolution of the screen	4065	729
getFontList()	Installed fonts	111	28
getTimezoneOffset()	Time Zone	23625	66329
offsetHeight, offsetWidth	Size of HTML element	74722	91362

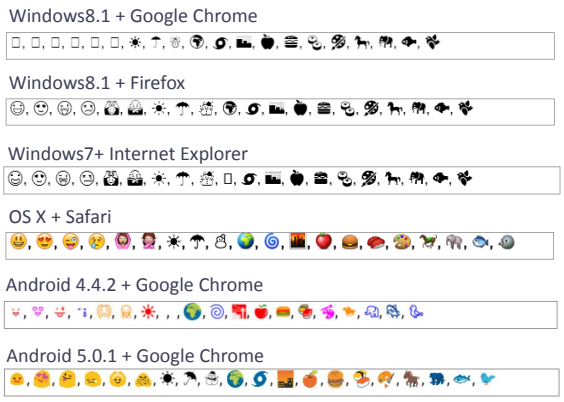

図 4: Canvas で絵文字を描画した時の環境ごとの違い.

$$H(X) = -\sum_{i=1}^{n} P(X = x_i) \log P(X = x_i) \quad (1)$$

Mowery ら [2] の実験では，294 種類のサンプルにおいて，CF のエントロピーは 5.73 bits となった．本システムで Canvas に描画する図形の種類には，テキスト，テキストの輪郭線 (Stroke)，四角形，円形，四角形のグラデーション，円形のグラデーションなどがある．テキストのうち，絵文字を Unicode で指定して描画したときの環境ごとの違いを図 4 に示す．Windows 環境では絵文字の色は白黒となり，ブラウザによって一部の絵文字が正しく表示されない．OS X では立体的なカラーの絵文字が表示され，Android では二次元のカラーの絵文字が表示される．このような違いによってユーザーの環境を識別することができる．

4.2 CSS Fingerprint

Canvas では任意の HTML の要素を指定して，スクリーンショットを撮るように Canvas 上に投影することができる．CSS3 では，要素の背景をグラデーション色にしたり，透過したり，影をつけたりなど，様々な表現の図形を描くことができる．本研究では，CF に加え，CSS で描いた図形を Canvas 要素を使って画像化した CSS Fingerprint を収集システムに組み込んだ．本システムで実際に使用した CSS の図形を図 5 に示す．また，HTML 要素の画像化の実装に際しては，html2canvas [17] というライブラリを使用した．

CF はユーザの OS やブラウザの種類によって値が変化するとされていたが，この CSS Fingerprint は，OS やブラウザの種類に加え，ブラウザのズーム表示の割合によって段階的に変化することが判明した．つまり，ユーザがサイトに訪れた瞬間に CSS Fingerprint を採取することで，ブラウザの既定のズーム値をある程度識別することが可能となる．本実験においては User-Agent (OS とブラウザの種類の組み合わせ) ごとに統計をとるため，他の Fingerprint との整合性をとるために，CSS Fingerprint 採取の際にはブラウザのズーム値を既定値から変更しないようにする．

図 5: CSS で描いた図形 (Windows8.1+Google Chrome).

4.3 Fingerprint の分析結果

4.1 で述べた Web サイトにおいて，計 85 種類の環境 (OS とブラウザの組み合わせ) におけるデータを収集した．本研究に用いた OS の内訳を表 7 に示す．

表 7: 分析対象の OS の内訳

OS	User-Agents	Rate %
Windows	41	48.2
Mac OS X	16	18.8
Linux	9	10.6
Android	12	14.1
iPhone OS	7	8.2

表 8: 収集した Canvas Fingerprint の分析結果

Graphic	Distinct Hashes	Entropy bits
Text (Arial Font)	40	4.96077
Text (No Real Font)	40	4.85240
Text (Small pixel)	34	4.63830
Text (Emoji)	59	5.71504
Stroke Text	34	4.46351
Rectangle	19	3.42149
Circle	32	4.43861
Rectangle Gradation	27	3.93603
Circle Gradation	24	3.62437
CSS	67	5.95922
All Combined	74	6.14169

また使用したブラウザは，Google Chrome，Internet Explorer，Firefox，Safari，Opera，Microsoft Edge などの主要なブラウザに加え，比較的シェア率の高いいくつかのブラウザも含まれている．これらの環境は Javascript の navigator.userAgent プロパティによって取得した User-Agent を元に識別する．

Canvas で描いた図形の種類と，ユニークなハッシュ値の数，User-Agent 毎のエントロピー値を表 8 に示す．それぞれの図形について，OS やブラウザによるハッシュ値のばらつきが見られた．最もエントロピーが高かったものは CSS Fingerprint であり，次に高かったものは図 4 で示した絵文字であった．また，これらの図形のハッシュ値をすべて連結して一つの Fingerprint としたところ，ユニークな Fingerprint の数は 74 となり，約 87%の割合でユーザの環境を一意に特定できることがわかった．この時のエントロピーは 6.14169 bits であり，Mowery ら [2] が提案した CF の採取手法を用いた時のエントロピー (5.73 bits) よりも高い結果となった．

一方で，ある異なる環境において同一の Fingerprint が採取されるケースが観測された．同一の Fingerprint が採取された環境の例を表 9 に示す．表 9 のように，OS のバージョンやブラウザのバージョンのわずかな違いがあったときに Fingerprint が一致するケースがある．グループ 1 については，iOS における Safari と Chrome は同じレンダリングエンジンを用いているために Fingerprint が一致した．グループ 2 については，Chrome と標準ブラウザの Fingerprint が一致しているが，これは Android 5.0 以降，標準ブラウザに Chrome を使用しているためである．

表 9: 同一の Canvas Fingerprint が採取される環境

Group	OS	Browser
1	iPhone OS 7.1.2	Chrome 44.0.2403.67
	iPhone OS 7.1.2	Safari 7.0
2	Android 5.0.1	Chrome 38.0.2125.509
	Android 5.0.1	標準ブラウザ (Chrome 39.0.0.0)
3	Windows 8.1	Chrome 44.0.2403.107
	Windows 8.1	Chrome 44.0.2403.130
4	OS X 10.10.2	Chrome 43.0.2357.134
	OS X 10.9.5	Chrome 43.0.2357.134

4.4 従来の Canvas Fingerprinting の検知手法について

本節では，第 2 章で述べた Acar ら [3] の CF の検知手法について，本実験の結果を元に考察を行う．分析の結果，Acer らが提案した検知手法は回避することが容易であることが分かった．Acar らが採用した Canvas を用いたトラッキングの検知条件は，以下の 3 つの項目が同時に当てはまることである．

1. 同一 URL 内に toDataURL メソッドと fillText(または strokeText) メソッドが呼び出されている．
2. Canvas で描画した画像の大きさが 16 × 16 pxel 以上である．
3. JPEG などの圧縮による劣化のある画像フォーマットを使用していない．

1. について，Canvas に描いた図形を画像データとして抽出する際に toDataURL メソッド用いてるが，他にも getImageData メソッドをもちいてピクセルの毎の色情報を取得することができる． 3.2 節で述べた Evercookie [15] は，この getImageData メソッドを用いて Canvas Fingerprint を採取している．また，fiilText メソッドや strokeText メソッドは Canvas 上にテキストを描画するメソッドであるが，本研究により，これらのメソッドを使わなくても，fillRect メソッドや arc メソッドといった図形を描画するメソッドを用いることで Fingerprint を採取できることがわかった．

次に 2. の条件は，画像のピクセル数が多いほど Fingerprint のエントロピーが大きくなるという仮説に基づくものである．本研究で 12 × 12 pixel の小さい画像を用いたところ，大きい画像と同様 User-Agent 毎にばらつきが見られた．エントロピーは大きな画像ほどではないものの，小さな画像をいくつも組み合わせることでエントロピーを増大させることができると推測できる．

最後に 3. について，画像フォーマットは toDataURL メソッドの引数で指定することができるが，引数なしの場合は PNG 形式になる．本研究で開発した Web システムにおいて，画像フォーマットに JPEG を指定することはなかったが，toDataURL メソッドの引数を “image/jpeg” に変えて追加実験を行ったとこ

ろ，PNG の場合と同様に User-Agent 毎にばらつきが見られ，Fingerprint として利用できることがわかった．

さらに，この検知条件では，3.2 節で述べた WordPress のスクリプトも検知の対象となってしまう．以上より，本研究によって，従来の CF 採取の検出手法は，False Positive，False Negative 共に生じてしまうことが判明した．したがって，今後はより網羅的な検出手法を確立していく必要がある．

5 議論

本章では，本研究の制限と今後の課題について述べる．クロールによる Web サイトの実態調査では，コードの静的解析により Web トラッキングのおおまかな傾向をつかむことができたが，厳密に Javascript の処理を追うことは難しい．たとえば，あるメソッドが実際に実行されるかどうか，Fingerprint が外部サーバに送信されているかを厳密に評価することはできない．また，静的解析においては JavaScript の圧縮，難読化に対応することは困難である．これらの課題を克服するための方法として，Web サイトで実際に JavaScript を実行し，動的解析を行うことが挙げられる．

もう一つの課題として，CF の検知手法の確立がある．Canvas は通常目的での利用とトラッキング目的での利用の区別が困難である．本研究によって従来の検知手法では不十分であることが分かったため，今後は動的な手法なども応用しながら，Canvas の画像データを抽出したり外部サーバへ送信したりする挙動などを捉える必要がある．

6 結論

本研究ではインターネット上の Web サイトで Canvas がどのような使われ方をしているのかを調査するために，良性サイトと悪性サイトの大規模なクロールを行った．その結果，良性サイトでは約 2%，悪性サイトでは約 0.3%の Web サイトが CF を採取している可能性が高いことがわかった．さらに，良性サイト，悪性サイトともに，CF を採取している Web サイトの 80%以上は，他の WBF も同時に採取していることがわかった．また，過去の実態調査からの大きな変化も見受けられ，今後 CF を用いた Web トラッキングが増加することも推測できる．

また，本研究では CF の特性を理解するために，Web サイトを作成し，CF の収集，分析を行った．その結果，Canvas に描画する図形の種類によって，ユーザーの環境ごとの Fingerprint のばらつきに違いがあることがわかった．さらに本研究で提案した CSS Fingerprint は他の Canvas 画像よりもばらつきが多く，ユーザーの識別により寄与することが分かり，今後対策を急ぐべきであることを示した．また，従来の CF の検知手法について評価を行った結果，その回避が容易であることを明らかにし，より網羅的な検知手法の必要性を示した．

参考文献

[1] P. Eckersley, “How unique is your web browser?,” in *Privacy Enhancing Technologies (PETs)*, 2010.

[2] K. Mowery and H. Shacham, “Pixel perfect: Fingerprinting canvas in HTML5,” in *Web 2.0 Workshop on Security and Privacy (W2SP)*, 2012.

[3] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz, “The Web Never Forgets: Persistent Tracking Mechanisms in the Wild,” in *In Proceedings of CCS 2014*, Nov 2014.

[4] K. Mowery, D. Bogenreif, S. Yilek, and H. Shacham, “Fingerprinting information in JavaScript implementations,” in *Web 2.0 Workshop on Security and Privacy (W2SP)*, 2011.

[5] M. Mulazzani, M. H. P. Reschl, M. Leithner, S. Schrittwieser, E. Weippl, and F. C. Wien, “Fast and reliable browser identification with JavaScript engine fingerprinting,” in *Web 2.0 Workshop on Security and Privacy (W2SP)*, 2013.

[6] T. Unger, M. Mulazzani, D. Fruhwirt, M. Huber, S. Schrittwieser, and E. Weippl, “SHPF: Enhancing HTTP(S) Session Security with Browser Fingerprinting,” in *In Availability, Reliability and Security (ARES)*, 2013.

[7] 磯侑斗, 桐生直輝, 塚本耕司, 高須航, 山田智隆, 武居直樹, and 齋藤孝道, “Web Browser Fingerprint を採取する Web サイトの構築と採集データの分析,” in **コンピュータセキュリティシンポジウム** *(CSS)*, 2014.

[8] D. Kravets, “Facebook’s $9.5 Million ’Beacon’ Settlement Approved.” `http://www.wired.com/2012/09/beacon-settlement-approved/`.

[9] Alexa Top Sites. `http://www.alexa.com/topsites`.

[10] AddThis. `http://www.addthis.com/`.

[11] URLBlacklist.com. `http://www.urlblacklist.com/?sec=home`.

[12] HTML Canvas Reference. `http://www.w3schools.com/tags/ref\_canvas.asp`.

[13] WordPress. `https://wordpress.org/`.

[14] Tanzina Vega, “New Web Code Draws Concern Over Privacy Risks.” `http://www.nytimes.com/2010/10/11/business/media/11privacy.html`.

[15] Samy Kamkar, “evercookie - virtually irrevocable persistent cookies.” `http://samy.pl/evercookie/`.

[16] G. Acar, M. Juarez, N. Nikiforakis, C. Diaz, S. Gurses, F. Piessens, and B. Preneel, “FPDetective: Dusting the Web for fingerprinters,” in *ACM Conference on Computer and Communications Security (CCS)*, 2013.

[17] html2canvas. `http://html2canvas.hertzen.com/`.

[18] G. Acar and M. Juarez and N. Nikiforakis and C. Diaz and S. Gurses and F. Piessens and B. Preneel, “New Web Code Draws Concern Over Privacy Risks.” `http://www.nytimes.com/2010/10/11/business/media/11privacy.html`.