Web Browser Fingerprint を採取する Web サイトの構築と採集データの分析

磯 侑斗† 桐生 直輝† 塚本 耕司† 高須 航†
山田 智隆† 武居 直樹† 齋藤 孝道‡

†明治大学大学院
214-8571 神奈川県川崎市多摩区東三田 1-1-1
{ce36004, ce36022, ce36032, ce46021, ce46035, ce46022}@meiji.ac.jp

‡明治大学
214-8571 神奈川県川崎市多摩区東三田 1-1-1
saito@cs.meiji.ac.jp

あらまし 今日の商業用の Web では広告事業者が利用者を追跡し，閲覧履歴に基づいた広告の提供を行うようになった．このような Web 行動追跡は HTTP クッキーを利用した方式が一般的だが，HTTP ヘッダや JavaScript から採取可能な情報（Browser Fingerprint）を用いることで端末が識別可能であることが示され，現在では Browser Fingerprint を用いた方式も一部で利用されている．我々は既存の Browser Fingerprint に加え，プライベート IP アドレスや HDD 空き容量など独自に導入したものも採取する Web サイトを開設し，Web Browser Fingerprint の収集を行っている．本論文では我々の Web サイトでの採取法を示し，採取したデータの分析結果を報告する．

An Implementation of Browser Fingerprinting Website and Analysis of Its Collected Data

Yuto Iso† Naoki Kiryu† Koji Tsukamoto† Ko Takasu†
Tomotaka Yamada† Naoki Takei† Takamichi Saito‡

†Graduate School of Meiji University
1-1-1, Higashimita, Tama-ku Kawasaki-shi, Kanagawa, 214-8571, Japan
{ce36004, ce36022, ce36032, ce46021, ce46035, ce46022}@meiji.ac.jp

‡Meiji University
1-1-1, Higashimita, Tama-ku Kawasaki-shi, Kanagawa, 214-8571, Japan
saito@cs.meiji.ac.jp

Abstract In recent years, online advertising companies are tracking their visiting viewers to provide advertisements responding to the user's browsing history. The common method of tracking viewers is to use HTTP Cookies. However, recent studies show that the browser can be identified by its characteristic called browser fingerprint and some companies started to use it. As an experiment, we launched a website to collect fingerprints which include ones we newly introduced. In this paper, we describe our fingerprinting website and report the analysis of collected fingerprints.

1 はじめに

今日の商業用の Web の利用では HTTP クッキーによるWebサイト利用者の識別や追跡が広く行われている. 特に, 第三者の広告事業者は利用者の行動に基づいた広告を配信する, 行動ターゲティング広告を実現するために複数の Web サイトを跨いだ行動追跡を行っている.

広告事業者が発行する HTTP クッキーは認証情報の管理などを目的としてWebサイトが発行する, いわゆる, ファーストパーティークッキーとは異なり, サードパーティークッキーとして区別している[18]. AppleのSafari[1]など一部のブラウザでは利用者のプライバシの保護を目的としてサードパーティークッキーをデフォルトで受け入れず, 他のWebブラウザでも設定によりサードパーティークッキーの拒否が可能である. そのためサードパーティークッキーの利用は特に制限されつつあると言える.

そのような中, Peter Eckersley により画面解像度やインストール済みフォントのリストなど, クライアント端末やブラウザの特徴となる情報, いわゆる, Web Browser Fingerprint を利用することにより 94.2%の利用者を識別できることが示された[3]. その後, 数多くの類似する研究が行われている.

既存の特徴に加え, 我々はプライベート IP アドレスや HDD の空き容量などを独自に導入し, Web Browser Fingerprint としての利用を提案した[19]. それに伴い Web サイトを開設し, Web Browser Fingerprint の採取を行っている. 本論文では我々の Web サイトでの採取法を示し, 採取したデータの分析結果を報告する.

2 Web Browser Fingerprinting

本章では, 現在使用されている Web Browser Fingerprint や近年の研究で新たに登場した Web Browser Fingerprint 及び Web Browser Fingerprinting への対策について説明する.

2.1 Web Browser Fingerprint

Web ブラウザが Web サーバへアクセスする際, Web サーバが取得できる Web ブラウザの情報を特徴点と呼ぶ. 特に, 端末やWebブラウザの利用者の特定につながるものを指す. 特徴点を 1 つ以上組み合わせたものを, Web Browser Fingerprint と呼ぶ. 本論文では, 以降 Fingerprint と呼ぶこととする. 特徴点の組合せから生成したハッシュ値や, 単一の特徴点を Fingerprint と呼ぶこともある.

Web ブラウザが Web サーバへアクセスした際に, Web サーバが Web ブラウザから Fingerprint を採取する行為を, Web Browser Fingerprinting と呼ぶ. 本論文では, 以降 Fingerprinting と呼ぶこととする.

Fingerprint は Web の広告事業者による行動追跡のほか, リスクベース認証や不正防止にも利用されている[21].

2.2 既存の Fingerprint に関する研究

Eckersley[3]は, Fingerprinting の研究を行い, JavaScript や Flash などから採取できる Fingerprint を用いて, 利用者の端末を 94.2% 一意に識別できることを示した. Eckersley が使用した特徴点には, UserAgent, HTTP Acept ヘッダ, ブラウザのプラグイン, タイムゾーン, スクリーンサイズ, フォント, HTTP クッキーの使用の可否, Supercookie に利用されるストレージの対応などがある.

Mowery[4]は, HTML5 の Canvas API や WebGL を用いて文字列や画像を描画し, 描画結果の画素レベルの差から利用者端末の GPU, OS 及び Web ブラウザの組合せが特定できることを示し, 300 個のサンプルを収集した際のエントロピーは 5.73 ビットであったとある. また, Mowery ら[5]は, JavaScript を用いたベンチマークを用いて, Web ブラウザの特定, OS や CPU の識別を行い, その実験結果を示している. Web ブラウザの識別に関しては 98.2%の確率で識別できるとしている.

Mulazzani ら[6]は, JavaScript エンジンの実装状況を特徴点として捉え, Web ブラウザの識別を行った. ここでの実装状況とは,

JavaScript エンジンが正しく実装されているか否かを指す.

著者らの研究グループでは, クライアント端末のハードウェア及びネットワークに関する特徴点の採取を行っており, 文献[7][8][9][15][19]では, CPU のコア数, GPU レンダリングの有無, メディアデバイスの有無や, タブレット端末は表示向きなどの情報の採取に成功している. また, [8]では, Mowery ら[5]のアプローチと同じとなるが, JavaScript の様々な処理のパフォーマンスによるベンチマークの採取により端末を推定し, それを特徴点とすることを独自に試みた. さらに, [9]では, CPU 拡張命令である SSE2 (Streaming SIMD Extensions 2)の対応の有無による CPU アーキテクチャの推測に成功している.

3 Web Browser Fingerprint 採取サイト

著者らの研究グループでは, 独自に導入した特徴点を含む, 端末のハードウェアやブラウザに関する情報を採取する Web サイト http://www.saitolab.org/fingerprint/ (以降 Fingerprint 採取サイトと呼ぶ)を立ち上げ Fingerprint の収集を行っている. 利用者はこのサイトで採取された Fingerprint とエントロピーを確認することができる. また, 関係者向けに採取の際に氏名やメモなどを同時に保存する機能も有する.

本節では Fingerprint 採取サイトでの採取フローや採取する特徴点について説明する.

3.1 採取フロー

Fingerprint 採取サイトの動作(以下, 採取フローという)図 1 により説明する. 図 1 の Web ブラウザと Web サーバ間の実線の矢印は画面遷移を伴う通信を表し, 破線の矢印は画面遷移を伴わない XHR(XMLHttpRequest)による通信を表す.

(1) 利用者が採取開始ボタンを押下すると Fingerprint を採取するページのリクエストが Web サーバへ送信される.

(2) Web サーバは(1)のリクエストに含まれる以下の情報をデータベースに保存する.
- 3.2 節に示す特徴点
- ブラウザ識別用の HTTP クッキー(以降 UID と呼ぶ)
- サーバの時刻

(1)のリクエストにブラウザ識別用の HTTP クッキーが存在しない場合は新たに生成しデータベースに保存する. また, ブラウザにも保存させる. この UID, は同じブラウザから複数回アクセスが有った場合にそのアクセスは同一のブラウザからのアクセスであることを, 判断するために利用する.

(3) Web サーバは Fingerprint を採取するページを送信する. このページには JavaScript と Flash を利用して特徴点を採取し, 送信する処理が記述されている. また, (7)でサーバから受け取る特徴点の値やエントロピーを表の形式で表示する処理も記述されている.

(4) Web ブラウザは 3.2 節に示す特徴点の一部を採取し, Web サーバへ XHR を利用して送信する. Web サーバは受信した特徴点の値とサーバの時刻をデータベースに保存する.
ただし, ハードディスクの空き容量やリフレッシュレートなどを取得する JavaScript の関数は非同期関数であるので, 通常はこれらの処理が同時に行われてしまう. 採取の際は互いに影響を及ぼさないよう, 逐次的に採取するようにしている.

(5) Web ブラウザは 3.2 節に示す特徴点の残りを(4)と同様に Web サーバへ送信する.
(4)と(5)の 2 回に分けて結果を送信する理由は, 時間の掛かる CPU ベンチマーク中にページが閉じられるなどして採取が中断された場合でも, 採取が完了した特徴点についてはデータベースに保存できるようにするためである.

(6) Web サーバはデータベースから Fingerprint を読み出し, 各特徴点のエントロピーを計算する.

(7) Web サーバは(6)で計算したエントロピーと

採取したFingerprintを返信する．Webブラウザは受信した情報を表示する．

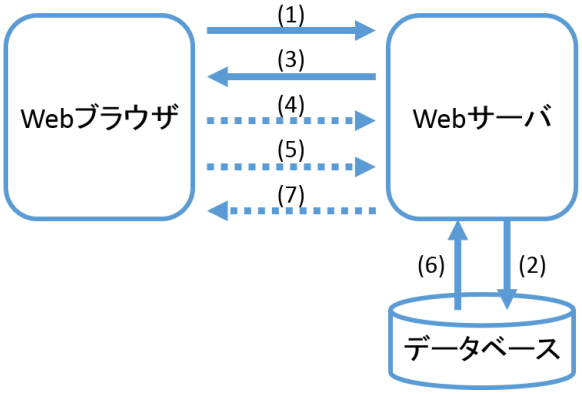

図 1 Fingerprint 採取サイトの採取フロー

3.2 採取する特徴点

Fingerprint 採取サイトで採取する特徴点を示す．

採取フローの(2)で採取する特徴点

- Accept Encoding や User Agent などリクエストに含まれるすべての HTTP ヘッダ
- OS Fingerprinting の結果
  p0f[14]を利用して採取する．
- グローバル IP アドレス
- リモートポート番号

採取フローの(4)で採取する特徴点

- User Agent
  JavaScript の navigator.userAgent から採取する．
- ローカルストレージの利用の可否
- セッションストレージの利用の可否
- 端末のタイムゾーン設定
  JavaScript の Date().getTimezoneOffset()から採取する．
- インストール済みプラグインの一覧
  navigator.plugins から採取する．プラグインの名前，説明，バージョンやファイル名が採取できる．
- インストール済みフォントの一覧
  Flash が利用できる場合は Action Script の Font.enumerateFonts で採取する．従来は取得したフォントリストの順序に決まりはなかったが Flash Player 14（2014 年 6 月リリース）からはソートされた結果が返るようになった[20]．Flash が利用できない場合，Internet Explorer では JavaScript の dlgHelper. fonts で採取が可能である．
- 画面解像度
  JavaScript の screen.width，screen. height と screen.colorDepth から採取する．また，Flash を利用することで Linux 環境ではデュアルモニタであるか否かの判定が行える．
- 画面の向き
  screen.orientation から採取する．このプロパティは現在標準化されておらず，ベンダプレフィクスをつける必要がある．Firefox の場合は mozOrientation，Internet Explorer の場合は msOrientation で採取できる．
- デバイスピクセル比
  window.devicePixelRatio から採取できる．スマートフォンではこの値が機種ごとに異なる．デスクトップのブラウザでは Web ページをズームすることによりこの値が変化する．
- 端末のタッチ機能の有無
  我々の研究グループが提案[15][19]した touch イベントを監視する方法で採取する．
- SSE2 対応の有無
  我々の研究グループが提案[9][19]した，JavaScript の計算誤差を利用する方式を利用する．採取する値は SSE2 対応の有無ではなく，計算結果自体である．計算結果は SSE2 の対応の有無のみではなく，ブラウザやそのバージョンによっても異なることが Fingerprint の分析から判明した．
- バッテリーステータス
  JavaScript の navigator.battery から採取する．
- ハードディスクの空き容量
  我々の研究グループが提案[15][19]した，Quota Management API を利用する方式で採取する．
- プライベート IP アドレスで示されている WebRTC を利用した方法[16][19]で採取する．

- MediaStreamTrack Device ID
  JavaScript の MediaStreamTrack.getSources()で取得できるカメラやマイクなどの識別子である.
- リフレッシュレート
  我々の研究グループが提案[15][19]した, Animation Timing API を利用する方式で採取する.
- 端末とサーバの時刻の差
- Canvas Fingerprinting の結果
  Mowery ら[4]によって提案された Canvas Fingerprinting の結果を採取する.

採取フローの(5)で採取する特徴点

- CPU ベンチマークの結果
  我々の研究グループが提案[7][19]した Web Workers を利用した CPU ベンチマークの結果を採取する.

4 採取した Fingerprint の分析

4.1 データセット

2013 年 12 月 6 日から 2014 年 8 月 14 日の期間で 1767 個の Fingerprint を収集した. この数には採取フローの(5)で採取する CPU ベンチマークが完了していないものも含まれ, それら除いた個数は 1667 個である.

収集したサンプルには 654 個の UID が含まれる. UID は利用者を識別するためにブラウザに保存した識別子であるので, 高々654 のブラウザからアクセスがあったことが分かる. このうち 266 個の UID から複数回のアクセスがあり, 前述のとおり 1767 個の Fingerprint を収集できた. 同一 UID での最初の採取から最後の採取までの期間が最長のものは 181 日であった. 30 日以上のものは 33 個, 7 日以上のものは 129 個であった.

4.2 節でのエントロピーの算出においては, サンプルの重複を防ぐために UID および Fingerprint が一致するサンプルは 1 つとカウントした. ここでの Fingerprint とは, 本論文で示す Fingerprint の中, 同一端末でも採取ごとに値が異なる, OS Fingerprinting の結果, リモートポート番号, バッテリーステータス, ハードディスクの空き容量, リフレッシュレート, 端末とサーバの時刻の差, GPU ベンチマークの結果と CPU ベンチマークを除いたものである. その結果, エントロピーの算出に使用するサンプルの数は 1228 個となった.

データセットに含まれるブラウザの割合を表 1 に示す. また, 参考として StatCounter[17]が提供する同時期の日本と世界でのブラウザのシェアも記載する. Fingerprint 採取サイトと他とで割合が異なるが, これらの原因としては, Fingerprint 採取サイトの利用者が CS 系の学生やこの分野に興味のある人に偏っているためであると考えられる.

表 1 データセットに含まれるブラウザの割合

Web ブラウザ	Fingerprint 採取サイト	StatCounter Japan	StatCounter Global
Chrome	36.8%	22.3%	37.5%
Firefox	25.8%	11.3%	14.0%
Internet Explorer	22.7%	34.1%	16.5%
Safari	3.2%	7.9%	7.5%
iPhone	3.1%	12.7%	5.3%
Android	1.5%	7.7%	7.1%

4.2 エントロピー

Fingerprint 採取サイトで収集した特徴点のエントロピーを表 2 に示す.

3.2 節で述べた特徴点のうち, OS Fingerprint, Canvas Fingerprint, MediaStreamTrack Device ID 及び画面の向きは, Fingerprint 採取サイト開設後に採取を始めた特徴点であるので十分なサンプル数を得られておらず, エントロピーの計算からは除外した. 前述のとおり, Fingerprint の収集開始時にはインストール済みフォントの一覧はソートされていなかったが, 3.2 節で示した通り, Flash Player のバージョンアップに伴い取得できるフォント一覧がソート済みとなった. [3]ではフォントの順序にも端末ごとの差が現れることから, 採取したフォントの順序も含めて比較しているが, この変更により順序の比較は意味を成さなくなった. そのため, ソートがされていない時期に採取したサンプルもソートし, エントロピーを算出した結果も記載した.

表 2 の“21 個の特徴点”は，表 2 の 21 個の特徴点（インストール済みフォントは未ソートのものを使用）を全て利用した場合のエントロピーである．“18 個の特徴点”は 21 個の特徴点から 4.3 節で述べる値が変わりやすい特徴点を除いたものである．

表 2 各特徴点のエントロピー

特徴点	エントロピー（bits）
インストール済みプラグイン	7.964657991
プライベート IP アドレス	7.003398756
UserAgent（JavaScript）	6.817763926
インストール済みフォント	6.696629821
インストール済みフォント（ソート）	6.518295724
UserAgent（HTTP ヘッダ）	6.512903442
グローバル IP アドレス	6.274712808
画面解像度	4.014056591
HTTP Accept Language	2.694879647
SSE テスト	2.493140583
HTTP Accept	1.895420708
デバイスピクセル比	1.445992639
HTTP Accept Encoding	1.298092836
HTTP Origin	1.014699729
HTTP Connection	0.845162779
タイムゾーン	0.404506546
タッチ機能の有無	0.401490172
HTTP Accept Charset	0.156397479
ローカルストレージ利用可否	0.124714106
セッションストレージ利用可否	0.105312710
HTTP Referer	0.031604031
HTTP クッキー利用可否	0
21 個の特徴点	10.15272696
18 個の特徴点（プライベート/グローバル IP とプラグインを含めない）	8.479666018

4.3 Fingerprint の推移

本節では同じ UID を持つサンプルを比較することにより，時間の経過に伴う特徴点の値の変化を観察する．

次の(1)，(2)式を満たす 2 つのサンプルの中，その 2 つのサンプルの特徴点の値が一致する割合を表 3 に示す．ここで，Xは収集した全サンプルの集合を表し，関数Uは入力のサンプルに対する UID への写像である．

$$x_1, x_2 \in X$$
$$U(x_1) = U(x_2) \qquad (1)$$

表 3 の期間は次の式を意味する．関数Tは入力としてのサンプルを採取した日付を返す．また，$n = 1,2,3,\ldots$とする．

$$t_n < |T(x_n) - T(x_{n+1})| \leq t_{n+1} \qquad (2)$$

例えば，表 3 の期間が 0～1（$t_0 = 0, t_1 = 1$）の場合，2 つのサンプルを取得した時刻の差が 0 日より大きく 1 日以内であることを表す．

画面の向き，画面解像度（Flash）及び WebRTC Device ID は他の特徴点と採取開始時期が異なるためサンプル数が異なる．

表 3 から，比較する 2 つのサンプルの採取時刻の差が 1 日以内であればどの特徴点も高い一致率となり，期間が長くなるにつれ一致率が低下する傾向があると分かる．[3]においてもインストール済みプラグインは最もエントロピーの高い特徴点であるとされているが，採取から 14 日以上経過すると 8 割以上のサンプルで値が変化している．つまり，採取から 14 日以上経過した後に端末を識別する場合，インストール済みプラグインを Fingerprint に含めると 8 割以上の端末が識別できないこととなる．インストール済みプラグインはバージョンアップが行われるなど，時間の経過に伴い変化しやすいので採取から 14 日以上経過した Fingerprint と比較する場合，この特徴点を Fingerprint に加えないか，変化しやすいバージョン番号などは考除いて比較する必要がある．また，プライベート IP アドレスとグローバル IP アドレスは 1 日以上経過すると半分程度が変化してするから，これらも Fingerprint には適さないことが分かる．

表 2 と表 3 には 21 個の特徴点からプライベート IP アドレス，グローバル IP アドレス及びインストール済みプラグインを除いた 18 個の特徴点のエントロピーと期間毎の一致率も記載している．21 個の特徴点を使った場合よりエントロピーでは劣るものの，より長い期間で多くのサンプルが一致している．

今回の分析により Fingerprint で端末を識別する場合，採取からの経過日数に応じて識別に用いる特徴点の最適な組合せが変化することが判明した．

4.4 同じ UID を持つブラウザ

Fingerprint の分析中に興味深い現象を発見したので報告する.

HTTP クッキーはブラウザごとに独立して保存されるので, 同じ端末に複数のブラウザが存在しても HTTP クッキーが共有されることは通常はない. Fingerprint 採取サイトで収集した Fingerprintも同じUIDを持つサンプルは, 同じベンダのブラウザから採取されたものであった. 例外として, Internet Explorer と Firefox から採取されたサンプルが同じ UID を持つケースが 1 件あった.

異なるブラウザが同じ UID を持つ原因として, User Agent が偽装されている場合が考えられるが, Firefox でのみ対応している API を利用して採取する特徴点が Firefox を名乗るブラウザでは採取でき, Internet Explorer を名乗るブラウザでは採取できないなどの点から User Agent の偽装が原因ではないと判断した. UID を手動でコピーした場合も考えられるが, 最も自然な理由として, 一度 Internet Explorer で Fingerprint 採取サイトを利用した後に Firefox で Internet Explorer のデータをインポートした場合が考えられる. ブラウザのインストール時などに, 他のブラウザからブックマークなどのデータを読み込むことができるが, このとき HTTP クッキーも別のブラウザから読み込み利用することができる. このサンプルはそのような環境で採取されたものであると考えられる.

5 まとめ

本論文では Fingerprint 採取サイトで採取している特徴点と採取フローを説明した. また, 収集した Fingerprint を分析することにより, 端末を識別する際に用いるべき特徴点がその最終時期に応じて変化することを示した.

今回分析に含めることができなかった, CPU ベンチマークの結果や HDD 空き容量などの分析は今後の課題である. また, 今回は, 変化しやすい特徴点を含めないことにより採取から時間が経過しても Fingerprint が一致することを示したが, 例えばプラグインのバージョンは考慮せずに比較するなど一致率を維持したままエントロピーを向上させる方法の考案も今後の課題である.

6 参考文献

[1] http://www.apple.com/jp/safari/

[2] http://samy.pl/evercookie/

[3] P. Eckersley, How Unique is Your Web Browser? In Proc. Privacy Enhancing Technologies Symposium (2010), LNCS vol. 6205

[4] K. Mowery and H. Shacham. "Pixel Perfect: Fingerprinting Canvas in HTML5." Web 2.0 Security and Privacy (W2SP). May 2012

[5] K. Mowery, D. Bogenreif, S. Yilek, and H. Shacham, Fingerprinting Information in JavaScript Implementations, In Web 2.0 Security and Privacy, 2011

[6] M. Mulazzani, P. Reschl and M. Huber, M. Leithner, S. Schrittwieser and E. Weippl, Fast and Reliable Browser Identification with JavaScript Engine Fingerprinting, in Web 2.0 Workshop on Security and Privacy (W2SP), 2013

[7] 後藤浩行, 齋藤孝道, Web 行動追跡のためのハードウェア特徴点の抽出, 2013 暗号と情報セキュリティシンポジウム概要集 p72-77

[8] 塚本耕司, 後藤浩行, 齋藤孝道, JavaScript ベンチマークを用いた CPU 推定手法の提案と実装, 第 75 回情報処理学会全国大会公演論文集 3-611, 3-612

[9] 桐生直輝, 後藤浩行, 齋藤孝道, CPU 拡張命令の対応の有無による CPU アーキテクチャの推測, 第 75 回情報処理学会全国大会公演論文集 3-613, 3-614

[10] https://www.ghostery.com/ja/

[11] http://fingerprint.pet-portal.eu/?menu=6

[12] https://www.torproject.org/projects/torbrowser.html.en

[13] http://www.saitolab.org/fingerprint/

[14] http://lcamtuf.coredump.cx/p0f3/

[15] 高須航，磯侑斗，桐生直輝，齋藤孝道，HTML5 APIにより取得可能なデバイス情報を利用した端末識別手法の提案と実装，第76回情報処理学会全国大会公演論文集 3-651，3-652
[16] http://net.ipcalf.com/
[17] http://gs.statcounter.com/
[18] 齋藤孝道,マスタリング TCP/IP 情報セキュリティ編,オーム社，2013
[19] 齋藤孝道，磯侑斗，桐生直輝，Web Browser Fingerprinting に関する技術的観点での一考察，2014 暗号と情報セキュリティシンポジウム概要集 p.9
[20] http://helpx.adobe.com/flash-player/release-note/fp_14_air_14_release_notes.html
[21] G. Acar, M. Juarez, N. Nikiforakis, C. Diaz, S. Gürses, F. Piessens, and B. Preneel, FPDetective: dusting the web for fingerprinters, in ACM SIGSAC Conference on Computer & Communications Security, 2013

表 3 同一 UID のサンプルにおける期間毎の一致率（一致率 70%以上は背景を灰色とする）

期間	0~1	1~7	7~14	14~21	21~28	28~35	35~42	42~49	49~
インストール済みプラグイン	94.39%	80.07%	72.54%	16.69%	7.37%	11.61%	20.86%	15.56%	4.97%
プライベート IP アドレス	95.42%	53.12%	44.83%	35.92%	36.97%	18.98%	28.78%	31.11%	11.18%
JavaScript UserAgent	94.81%	90.22%	87.02%	73.12%	59.35%	34.56%	28.06%	24.44%	2.48%
インストール済みフォント	96.54%	90.71%	89.84%	82.42%	79.89%	70.54%	68.35%	55.56%	72.05%
インストール済みフォント（ソート）	96.54%	91.62%	91.28%	88.53%	88.81%	86.40%	76.25%	60.00%	72.04%
HTTP UserAgent	94.68%	89.79%	87.02%	73.50%	59.35%	34.56%	30.22%	24.44%	4.35%
グローバル IP アドレス	94.62%	42.18%	34.44%	28.54%	12.32%	13.88%	20.86%	31.11%	9.94%
画面解像度	97.95%	94.38%	92.61%	92.36%	91.78%	91.78%	92.81%	84.44%	82.61%
HTTP Accept Language	98.22%	99.76%	99.64%	99.75%	99.72%	98.30%	100.00%	100.00%	100.00%
SSE テスト	99.79%	100.00%	99.94%	99.87%	100.00%	99.72%	99.28%	97.78%	73.91%
HTTP Accept	96.97%	99.76%	99.58%	99.75%	99.72%	98.02%	100.00%	100.00%	96.89%
デバイスピクセル比	98.43%	99.82%	99.22%	96.05%	96.03%	92.92%	86.33%	66.67%	68.94%
HTTP Accept Encoding	97.66%	98.04%	99.52%	99.36%	99.58%	98.30%	100.00%	100.00%	100.00%
HTTP Origin	98.27%	100.00%	100.00%	100.00%	100.00%	99.72%	100.00%	100.00%	100.00%
HTTP Connection	97.53%	99.88%	99.82%	99.87%	99.86%	99.15%	100.00%	100.00%	100.00%
タイムゾーン	99.97%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
タッチ機能	99.95%	100.00%	99.82%	99.87%	100.00%	100.00%	100.00%	100.00%	100.00%
HTTP Accept Charset	98.64%	99.63%	99.82%	99.87%	99.86%	99.15%	100.00%	100.00%	100.00%
ローカルストレージ	99.68%	99.88%	100.00%	100.00%	100.00%	99.72%	100.00%	100.00%	100.00%
セッションストレージ	99.73%	100.00%	100.00%	100.00%	100.00%	99.72%	100.00%	100.00%	100.00%
HTTP Referer	99.79%	99.14%	99.58%	99.62%	99.86%	99.15%	100.00%	100.00%	100.00%
HTTP クッキー	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
21 個の特徴点	85.41%	15.22%	7.39%	2.67%	0.84%	0.0%	0.0%	2.22%	0.0%
18 個の特徴点	92.28%	85.94%	86.59%	82.16%	83.71%	76.48%	61.87%	22.22%	32.29%
比較対象となる組の数	3758	1636	1664	785	706	353	139	45	161
画面の向き	99.72%	99.68%	99.91%	100.00%	100.00%	-	-	-	-
比較対象となる組の数	3249	1290	1172	351	28	0	0	0	0
画面解像度（Flash）	97.46%	93.26%	91.86%	91.45%	88.88%	100.00%	-	-	-
比較対象となる組の数	3355	1367	1316	433	171	1	0	0	0
WebRTC Device ID	98.71%	99.61%	99.74%	100.00%	100.00%	-	-	-	-
比較対象となる組の数	3259	1296	1172	351	28	0	0	0	0