Computer Security Symposium 2017
23 - 25 October 2017

Browser Fingerprintingにおける特徴の組み合わせに関する考察

田邉 一寿[1]　高橋 和司[1]　安田 昂樹[1]　種岡 優幸[1]　細谷 竜平[1]　小芝 力太[2]　齋藤 祐太[2]
齋藤 孝道[2]

概要：端末から採取可能な複数の特徴点を用いて端末内のブラウザを Web サーバが識別する Browser Fingerprinting と呼ばれる手法がある．既存研究において，さまざまな特徴点が提案されたが，特徴点の組み合わせと識別精度の相関関係については議論されてこなかった．本論文では，我々の研究グループが運営する Web サイトで採取した 9,457 サンプルを用いて，特徴点の組み合わせにおける精度を算出し，識別精度が最良となる組み合わせを OS ごとに求めた．その結果，Panopticlick [1] が使用していた特徴点を用いた場合と比較し，Windows では 1.006%，Mac では 3.894%，iOS では 7.456%，Android では 3.840%の精度の向上を実現した．

キーワード：Browser Fingerprinting，Web 行動追跡，Web 技術

Study on Combination of Features in Browser Fingerprinting

Kazuhisa Tanabe[1]　Kazushi Takahashi[1]　Koki Yasuda[1]　Masayuki Taneoka[1]　Ryohei Hosoya[1]
Rikita Koshiba[2]　Yuta Saito[2]　Takamichi Saito[2]

Abstract:
Browser Fingerprinting is a technique which identifies browsers in the device by using device features that can be collected by Web servers. Various features have been proposed in the previous research; however, the correlation between the feature combination and the identification accuracy has never been discussed. In this paper, we calculated the identification accuracy of each feature combination by using 9,457 samples collected on our website. As a result, compared to Panopticlick[1], we realized 1.006 % on Windows, 3.894 % on Mac, 7.456 % on iOS, and 3.840 % on Android improvement of identification accuracy.

Keywords: Browser Fingerprinting, Web Targeting, Web Technique

1. はじめに

端末から採取可能な複数の特徴点を用いて Web サーバが端末内のブラウザを識別する Browser Fingerprinting と呼ばれる技術が普及している．Browser Fingerprinting は，Browser Fingerprinting を行う Web サーバへ接続するブラウザの IP アドレスや UserAgent 文字列など，接続に伴って Web サーバが採取できる情報を特徴点として，その組み合わせなどにより，当該ブラウザを特定する．

Browser Fingerprinting は，Web 広告事業者による行動追跡やリスクベース認証に利用されている．2016 年 1 月に行われた調査 [4] によると，Alexa [10] のランキング上位 100 万サイトのうち約 1.6%が Fingerprinting を行なっており，さらに上位 1,000 サイトのうち約 5.1%が Fingerprinting を行なっていることが既に判明している．

既存研究において，さまざまな特徴点が提案されたが，特徴点の組み合わせと識別精度の相関関係については議論

1 明治大学大学院
Graduate School of Meiji University
2 明治大学
Meiji Univevrsity

されてこなかった．特徴点の中には時間経過に伴って値が変化してしまい識別精度を低下させるもの [7] や，多くの端末で同じ値を取ってしまい識別精度にあまり影響を与えないものがある．

そこで，本論文では，現状知られている特徴点の組み合わせとその識別精度を網羅的に調べ，その相関関係を求めた．識別精度は，我々の研究グループが運営する Web サイトで採取した 9,457 サンプルから算出した．その結果，識別精度を向上させる特徴点と，反対に低下させる特徴点があることが明らかになった．調査で明らかになった識別精度を低下させる特徴点を除去し，識別精度の向上を図ったところ，Panopticlick [1] と比較して，2.527%，AmIUnique [2] と比較して 0.834%の精度の向上を実現した．

また，OS ごとにサンプルを分類して，それぞれにおいて識別精度が最良となる特徴点の組み合わせを調べた．その結果，Panopticlick [1] が使用していた特徴点を用いた場合と比較し，Windows では 1.006%，Mac では 3.894%，iOS では 7.456%，Android では 3.840%の精度の向上を実現した．

2. Browser Fingerprinting

特徴点の組み合わせによって端末内のブラウザを識別する手法を Browser Fingerprinting（以降，Fingerprinting と呼ぶ）という．また，Fingerprinting に使用する特徴点の組み合わせを Browser Fingerprint（以降，Fingerprint と呼ぶ）という．UserAgent 文字列のように HTTP リクエストヘッダから採取できるものと，画面解像度のように JavaScript や CSS を用いて採取するものがある．

2.1 関連研究

Eckersley [1] は Fingerprint を採取するサイトを構築し，Fingerprint の分析をした．その結果，採取した Fingerprint の 83.6%がユニークであり，Flash や Java が実行できる端末に限定した場合，94.2%がユニークであることを示した．

Mowery ら [3] は，Canvas API を用いて端末内のブラウザ上に複雑な文字や図形を描写した結果が GPU，ブラウザのバージョン，及び OS の組み合わせによって異なることを示した．また，この調査の中で採取した 294 個のサンプルにおいて，116 個がユニークであり，この結果をハッシュ化して保存することで，Fingerprinting に応用できることを示した．そのときのエントロピーは 5.73 ビットであり，98.2%の確率でブラウザを正しく識別できるとしている．また WebGL API を用いて GPU やモデル名が採取可能であることを示した．

Laperdrix ら [2] は，Panopticlick [1] が使用した 10 個の特徴点に加え，HTTP ヘッダーのリスト，Ad Block 拡張機能の使用状況，OS の情報，Do Not Track の値，Mowery ら [3] の示した Canvas Fingerprinting，WebGL API から得られる GPU のベンダとモデル名について，それぞれ考察した．

2.2 識別精度の評価手法

Fingerprinting においての識別とは，同一の端末内のブラウザから採取したサンプルを同一とみなし，かつ，異なる端末内のブラウザから採取したサンプルを異なるとみなすことである．本論文では， ROC（Receiver Operating Characteristic）曲線の AUC（Area Under the Curve）を識別精度として定義する．AUC は ROC 曲線下の面積であり，識別精度が高いほど AUC は高い値をとる．ROC 曲線の AUC の値を識別精度として採用することで，同一の端末内のブラウザから採取した Fingerprint を同一とみなした割合である TPR（True Positive Rate）と，異なった端末のブラウザから採取した別の Fingerprint を同一とみなした割合である FPR（False Positive Rate）を共に確認できる．

2.3 正規化エントロピー

一般的には，Shannon Entropy を用いて，特徴点がもつ情報量を数値化することが多い．しかしながら，Shannon Entropy は，サンプル数によって値が変化してしまう．よって，本論文では式 (1) に示す Normalized Shannon's Entropy（以降，NE と呼ぶ）により，特徴点の情報量を表す．また，N はサンプル数を表す．

$$NE = \frac{-\sum_{A \in \Omega} P(A) \log_2 P(A)}{\log_2 N} \qquad (1)$$

3. サンプルの採取と分析の手法

3.1 Fingerprint 採取サイト

我々の研究グループでは，Fingerprint を採取する Web サイト（以降，Fingerprint 採取サイトと呼ぶ）を公開し，アクセスした端末の Fingerprint を採取している．採取した Fingerprint は，それと紐付けて生成した HTTP クッキー（以降，UID と呼ぶ）と併せて，Fingerprint 採取サイトのデータベースに保存される．この UID はアクセスした端末のブラウザにも保存され，同じ端末のブラウザから複数回アクセスがあった場合にそのアクセスを同一の端末のブラウザからのアクセスであることを判定するために利用している．なお，Fingerprint 採取サイトは利用者の同意の上で，Fingerprint を採取している．

3.2 実験に使用したサンプル

本論文では，2013/12/6 から 2017/8/2 までの期間に採取された 9,457 サンプルを用いて解析を行った．識別精度を求めるにあたり，一回のみアクセスのあったサンプルを含めると FPR の値を増加させてしまうので，複数回アク

セスのあったサンプルのみを実験に利用した．なお，UID数は 1,588 個だった．

Windows，Mac，Android，iOS の 4 つの OS に分類したサンプル数を表 1 に示す．ただし，OS の種類は UserAgent 文字列から抽出しているので，UserAgent 文字列が偽装されていた場合，誤分類される可能性があることに注意されたい．また，4 つの分類に該当しない OS と判定されたサンプルも存在した．具体的には，UserAgent 文字列から Ubuntu と判定されたサンプルが 35 サンプル，Debian と判定されたサンプルが 11 サンプル，Gentoo と判定されたサンプルが 2 サンプル，その他の Linux と判定されたサンプルが 51 サンプル，その他と判定されたサンプルが 16 サンプル存在した．

表 1 OS ごとのサンプル数

OS	Windows	Mac	Android	iOS	合計
サンプル数	4,087	1,245	2,508	1,483	9,323

3.3 特徴点の組み合わせの決定

本論文では以下の（1）〜（3）を再帰的に繰り返し，識別精度が最良になる特徴点の組み合わせを計算した．ただし，終了条件は，（1）で使用する特徴点が 2 個になった場合である．

(1) 特徴点 x 個の中から 1 つを除く組み合わせ ${}_xC_{x-1}$ を求める
(2) （1）で求めたすべての組み合わせにおける AUC の値を計算する
(3) （2）の中で AUC の値が最も高くなった特徴点の組み合わせを使って（1）〜（3）を再帰的に実行する

4. 分析に使用した特徴点

本節では，本論文の分析に使用した特徴点の中で，我々の研究グループが独自に採取している特徴点を中心に解説する．

4.1 math.tan の値

JavaScript において OS やブラウザに依存して浮動小数点演算に誤差が生じることが知られている [11]．本研究では引用文献で示されている math.tan(-1e300) を用いた．2017 年 6 月 15 日から採取を開始した特徴点であり，サンプル数は 225 個である．

4.2 SSE2 が利用可能か否か

CPU の拡張命令である，Streaming SIMD Extensions 2 (SSE2) が有効であるかを識別する手法が先行研究 [5] において示されている．先行研究 [5] では，この情報から CPU が Pentium 以降の CPU であるかどうかを推定していたが，本論文では，演算の結果を特徴点として用いた．

4.3 GPU のベンチマーク

先行研究 [7] において GPU ベンチマークを特徴点とする手法が提案された．我々の研究グループでは，Canvas の領域に繰り返し 1 ピクセルの正方形を 100,000 回描写した時にかかる時間と，1280 × 960 ピクセルの長方形を 100,000 回描写した時にかかる時間を連結させた値を特徴点としている．

4.4 Web Workers

先行研究 [6] にて，CPU の処理性能の差を利用して，利用者の端末の CPU のコア数を推定する手法が提案された．本論文では，Web Workers を利用して並列処理を実行させた結果のベンチマークの値を特徴点として利用した．

4.5 タッチ機能の有無

スマートフォンや一部の PC にはタッチ機能が搭載されている．JavaScript の Touch Event の ontouchstart により，端末にタッチ機能が搭載されているかを調べることができる．また，同時にタッチすることのできる数も採取できる [9]．本論文では，タッチ機能の有無のみ特徴点として採取している．2017 年 8 月現在，Safari と IE，Opera では使用することができない．

4.6 p0f による推定値

TCP のパケットから OS の種類や設定言語を推定する OS Fingerprinting という手法がある [13]．Fingerprint 採取サイトでは，この手法により推定される値を特徴点として採取している．

4.7 Fingerprint の文字列連結

我々の研究グループでは，20 個の特徴点を文字列連結してハッシュ化した値を保存しており，その値を特徴点としている [8]．使用している特徴点は，グローバル IP アドレス，HTTP Accept，HTTP Accept Charset，HTTP Accept Encoding，HTTP Accept Language，HTTP Connection，HTTP Origin，UserAgent 文字列，リファラ，タッチ機能の有無，タイムゾーン，JavaScript から採取される UserAgent 文字列，セッションストレージの利用可否，ローカルストレージ，画面解像度，デバイスピクセル比，プラグインリスト，SSE2 が利用可能か否か，プライベート IP アドレス，フォントリストの計 20 個である．

4.8 Tor Exit Node の IP アドレスであるか

Tor Project では Tor Exit Node の IP アドレスを公開している [14]．我々の研究グループが運用するサーバでは

Tor Exit Node の IP アドレスリストを毎日一回更新し，それと一致する IP アドレスを 1，不一致を 0 としたものを特徴点としている．今回使用したサンプルの内，9 つのサンプルにおいてこの特徴点の値が 1 となっていた．

4.9 Google Geolocation API

Google 社が提供する Google Map Geolocation API [12] の json 形式のレスポンスを特徴点としてデータベースに格納している．アクセスしてきたブラウザで実行されることで位置情報を得る．これを特徴点としてデータベースに格納する．

4.10 ブラウザの種類（ファミリ）

UserAgent 文字列からブラウザの種類を抽出した．サンプルにおけるブラウザの割合を図 1 に示す．本論文で使用したサンプルにおいて 28 種類のブラウザが観測された．

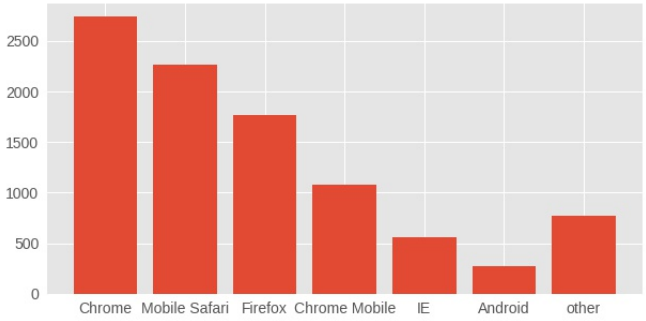

図 1 サンプルにおけるブラウザの割合

5. 実験の結果

5.1 識別に利用する特徴点の数と AUC の値の関係

識別に利用する特徴点の数と，AUC の値の関係を図 2 に示す．この図から，識別に利用する特徴点の数を増やしていくとある値で AUC の値が最大になり，その後は特徴点の数を増やしても AUC の値は減少することが分かった．すなわち，特徴点をむやみに増やすと，識別に悪影響があることがわかる．

5.2 識別精度が最良となる特徴点の組み合わせ

Fingerprint 採取サイトで採取された 9,457 サンプルを用いて，識別精度が最良となる特徴点の組み合わせを算出し，その結果を表 A·1 にまとめた．また，特徴点の組み合わせを先行研究と比較するために，表 A·1 には Panopticlick [1] と AmIUnique [2] が使用した特徴点も記載した．

図 3 に，先行研究 [1] [2] で使用した特徴点による組み合わせと，本論文において最良とした特徴点の組み合わせの ROC 曲線を示す．実験の結果，本論文で選択した特徴点の組み合わせを用いて計算した AUC の値は，Panopticlick [1] と比較して 2.527%，AmIUnique [2] と比較して

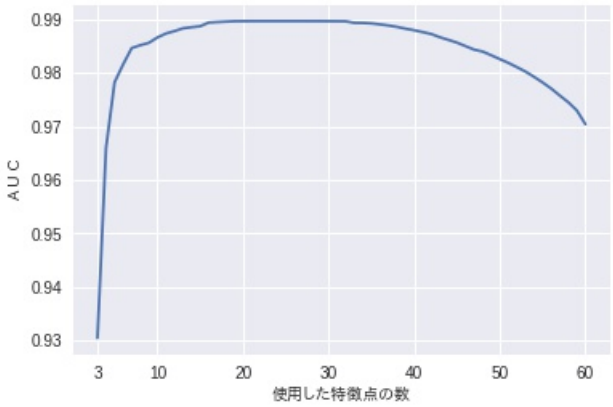

図 2 使用した特徴点の数と AUC の値

0.834%それぞれ高くなった．なお，今回使用したサンプルでは，Panopticlick [1] が採取している SuperCookie と AmIUnique [2] が採取している Use of ad blocker の特徴点を採取していない．

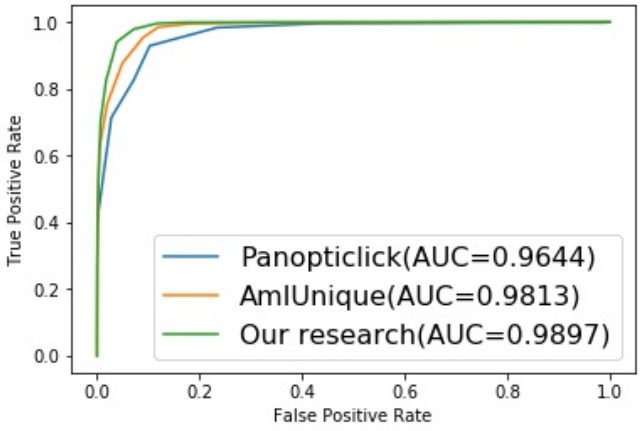

図 3 先行研究と本論文の ROC 曲線

5.3 OS ごとに分類した場合

OS ごとに識別精度が最良となる特徴点の組み合わせを計算した．分類した OS の種類は，Windows，Mac，iOS，及び，Android の 4 つである．実験の結果を表 2 にまとめた．このなかで，UserAgent 文字列，HTTP Connection，画面解像度，p0f による推定値はすべての OS で採用された．

結果として，Panopticlick [1] が使用していた特徴点を用いた場合と比較し，Windows では 1.006%，Mac では 3.894%，iOS では 7.456%，Android では 3.840%の精度の向上を実現した．

OS を区別しない状態で最良となった特徴点の組み合わせと，OS ごとに算出した最良の特徴点の組み合わせを比較するために，OS ごとの ROC 曲線を図 4，図 5，図 6，図 7 に示した．

表 2　OS ごとの最良の特徴点の組み合わせ

特徴点	Windows	Mac	Android	iOS
UserAgent 文字列	✓	✓	✓	✓
HTTP Connection	✓	✓	✓	✓
画面解像度	✓	✓	✓	✓
p0f による推定値	✓	✓	✓	✓
フォントリスト	✓	✓	✓	
リクエストヘッダ	✓	✓	✓	
メディアストリームトラック	✓	✓	✓	
Web GL	✓	✓		✓
HTTP クッキーの利用可否	✓		✓	✓
Do Not Track		✓	✓	✓
ブラウザの種類		✓	✓	✓
HTTP Accept	✓	✓		
プライベート IP アドレス	✓	✓		
プラグインリスト	✓		✓	
Tor の IP アドレス	✓		✓	
HTTP Accept Language	✓		✓	
デバイスピクセル比	✓		✓	
hardware_concurrency	✓		✓	
GPU	✓			✓
Canvas	✓			✓
Google Geolocation API	✓			✓
HTTP Origin	✓			✓
Web GL Renderer		✓	✓	
HTTP Accept Encoding		✓		✓
グローバル IP アドレス			✓	✓
worker	✓			
タッチの有無	✓			
HTTP Accept Charset	✓			
バッテリーの状態	✓			
近接センサ		✓		
タイムゾーン			✓	
ローカルストレージ			✓	
カメラ				✓

6. 議論

6.1 価値の低い特徴点

表 A·1 から，先行研究 [1] [2] で使用されていたタイムゾーン，プラグインリスト，HTTP Accept Encoding，HTTP Accept Language は，識別において価値がなかったことが示された．

以降で，先の 4 つの特徴点が，最良の特徴点の組み合わせから除かれた理由を考察する．

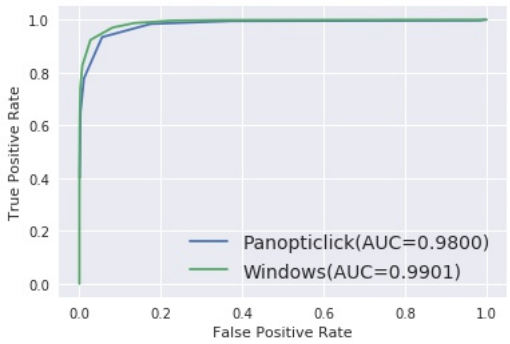

図 4　Windows における最良の特徴点の組み合わせの比較

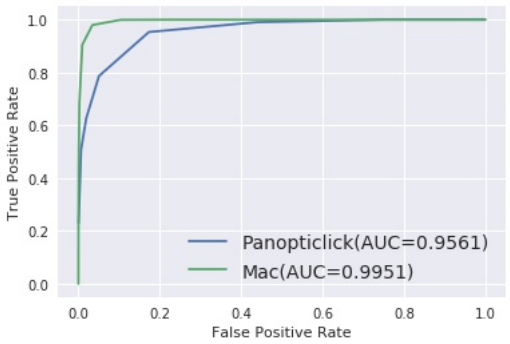

図 5　Mac における最良の特徴点の組み合わせの比較

6.1.1 タイムゾーン

我々の採取したサンプルの 97.7%が同一のタイムゾーンであることがわかった．タイムゾーンとそのサンプル数を表 3 に示す．本論文の実験で使用したサンプルでは 26 種類のタイムゾーンが観測された．

表 3　タイムゾーンのデータ分布

タイムゾーン	サンプル数
9	9,235
0	61
-7	59
1	24
-8	18
その他	59

タイムゾーンの NE を先行研究 [1] [2] と比較し，表 4 に示す．表 4 から，本論文で使用したサンプルは先行研究 [1] [2] と比較して，NE の値が低いことがわかる．これにより，タイムゾーンが最良の特徴点から除かれたことがわかる．

表 4　タイムゾーンのエントロピーの比較

	Panopticlick	AmIUnique	本論文
timezone	0.201	0.161	0.018

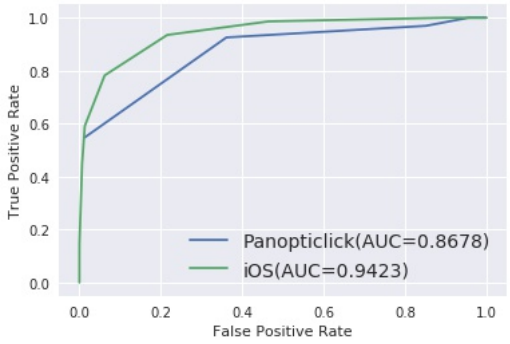

図 6　iOS における最良の特徴点の組み合わせの比較

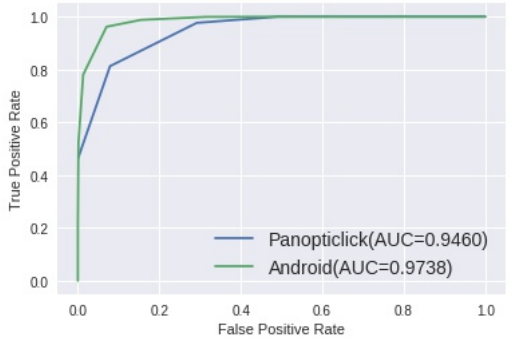

図 7　Android における最良の特徴点の組み合わせの比較

6.1.2 プラグインリスト

先行研究 [7] において，プラグインリストは時間経過にともなって変化しやすく，14 日後には一致率が 16.09%まで低下することが明らかにされている．今回の実験に用いたサンプルにおいて，59.1%が 14 日以上のアクセス間隔で採取されていることが分かった．

この結果，サンプルのプラグインリストが時間経過にともなって変化したことが原因で，プラグインリストが最良の特徴点から除かれたと推測できる．

6.1.3 HTTP Accept Encoding

HTTP Accept Encoding の値は HTTP リクエストヘッダに包含されており，HTTP Accept Encoding 単体で特徴点とする場合より，HTTP リクエストヘッダを特徴点とすることでより良い AUC の値をとることがわかる．

6.1.4 HTTP Accept Language

HTTP Accept Language の値は HTTP リクエストヘッダに包含されており，HTTP Accept Language 単体で特徴点とする場合より，HTTP リクエストヘッダを特徴点とすることでより良い AUC の値をとることが分かる．

6.2 HTTP クッキーの利用可否

HTTP クッキーの利用可否の値が False となっているサンプルがあった．本論文では，Fingerprint 採取サイトで採取されたサンプルのなかで複数回アクセスのあったサンプルのみを分析の対象としており，同一端末からのアクセスかどうかは UID（HTTP クッキーに保存された端末ごとに一意な値）の値を確認して判断している．

HTTP クッキーが利用できない場合，仮に同じ端末であったとしてもアクセスごとに UID の値が変化してしまう．したがって，本来であればすべてのサンプルにおいて HTTP クッキーが利用できるはずである．

本論文で使用したサンプルにおいて，HTTP クッキーの利用可否が False になっている 18 個のサンプルはすべて同じ UID をもつ同一利用者からのものであった．この UID は UserAgent 文字列や画面解像度がアクセスごとに変化しており，研究グループ内部の人間によるテストデータであることが推測される．この UID を持つサンプルを除いた場合，HTTP クッキーの利用可否はすべてのサンプルが True になる．よって，今回のような実験では，HTTP クッキーの利用可否は識別精度が最良になる特徴点の組み合わせから除かれる．

6.3 UserAgent 文字列の偽装検知

本論文での実験では，UserAgent 文字列の偽装は想定していない．しかし，UserAgent 文字列の偽装が疑われるサンプルが判明したので，その過程を示す．

iOS では通常，プライベート IP アドレスは採取できない．これはプライベート IP アドレスを採取する際に利用

する WebRTC が iOS では利用できないからである．しかし，iOS に分類された 2 つのサンプルにおいてプライベート IP アドレスが採取されていた．画面解像度を確認したところ，iPhone や iPad とは異なる画面解像度の値が採取されおり，UserAgent 文字列が偽装されている可能性が高い．なお，この 2 つのサンプルは同一の UID であり，同じ端末からのアクセスである．今回の分析により，画面解像度などの OS ごとに固有の値を用いることで，UserAgent 文字列の偽装を検知できる可能性があることがわかった．

6.4 採取時間の短縮

採取時間の短縮について調査した結果を報告する．この調査のために，Fingerprint 採取サイトと同じ端末 (Ubuntu16.04) 上に，本論文で示した最良の特徴点のみを採取するサイト（以降，最適化した Fingerprint 採取サイトと呼ぶ）を作成した．Fingerprint 採取サイトと最適化した Fingerprint 採取サイトにおいて，採取を開始するボタンをクリックしてから採取が完了するまでの時間を OS ごとに計測した．この調査の結果，Windows では 17.0%，Mac では 11.7%，iOS では 11.5%，Android では 5.0%の採取時間の短縮を確認した．なお今回の調査では，採取が完了するまでの時間がキャッシュの影響を受けないように，Chrome のプライベートモードを用いた．また，各アクセスごとに若干の誤差が生じる可能性があるので，5 回アクセスした平均値をとった．

6.5 Fingerprint の保存サイズの削減について

ここで，Fingerprint のサイズの削減について議論する．Fingerprint 採取サイトで採取している特徴点は 60 個である．すべて特徴点の値を文字連結したとき，1 サンプルあたりの文字数の平均は 1,304 文字だった．仮に本論文における最良の特徴点の組み合わせのみ採取した場合，採取する特徴点の数は 19 個になり，すべての特徴点の値を文字連結したときの文字数の平均は 368 文字になる．これにより，採取する特徴点を選別することにより，保存する Fingerprint のサイズを約 72%削減できる．なお，UserAgent 文字列などのデータ量の多い傾向のある特徴点を整形せずに保存することが想定されるが，実用的に Fingerprinting を行う場合，データ量の多い特徴点の値をハッシュ化し，サイズを圧縮してからデータベースに保存することも可能である．

7. まとめ

本論文では，我々の研究グループが運営する Web サイトで採取した 9,457 サンプルを用いて，特徴点の組み合わせにおける精度を算出し，識別精度が最良となる特徴点の組み合わせを求めた．その結果，Panopticlick [1] と比較して，2.527%，AmIUnique [2] と比較して 0.834%の AUC の値の向上を実現した．また，OS ごとに最適な特徴点の組み合わせを示し，Panopticlick [1] が使用していた特徴点を用いた場合と比較し，Windows では 1.006%，Mac では 3.894%，iOS では 7.456%，Android では 3.840%の精度の向上を実現した．今回の論文の成果により，特徴点を選別することで，以下の 2 つの成果を得た．

Fingerprint 採取時間の短縮　我々の研究グループが採取している 60 個の特徴点を採取した場合と比較して，最良の特徴点のみを採取した場合，Windows では 17.0%，Mac では 11.7%，iOS では 11.5%，Android では 5.0%の採取時間の短縮を確認した．

Fingerprint 保存サイズの削減　我々の研究グループが採取している 60 個の特徴点を採取した場合と比較して，最良の特徴点のみを採取した場合，Fingerprint 保存サイズを約 72%削減できることが明らかになった．

参考文献

[1] P. Eckersley, "How Unique Is Your Web Browser?", in Proc. of the 10th international conference on Privacy enhancing technologies (PETS'10), 2010

[2] P. Laperdrix, W. Rudametkin and B. Baudry, "Beauty and the Beast: Diverting modern web browsers to build unique browser fingerprints", in Proc. of 37th IEEE Symposium on Security and Privacy (S&P'16), 2016

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

[4] S. Englehardt and A. Narayanan, "Online Tracking: A 1-million-site Measurement and Analysis", in Proc. of ACM Conference on Computer and Communications Security (CCS'16), 2016

[5] 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, 2013

[6] 桐生直輝, 磯侑斗, 金子洋平, 齋藤孝道, "Web Workers を用いた演算処理性能の差による CPU コア数の推定", コンピュータセキュリティシンポジウム 2014 (CSS'14), 2014

[7] 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, "Web Browser Fingerprint を採取する Web サイトの構築と採取データの分析", コンピュータセキュリティシンポジウム 2014 (CSS'14), 2014

[8] 石川貴之, 高須航, 山田智隆, 武居直樹, 細井理央, 安田昂樹, 高橋和司, 齋藤孝道, "Fuzzy Hashing を用いた比較による長期的な Browser Fingerprinting の端末識別手法の提案", コンピュータセキュリティシンポジウム 2015 (CSS'15), 2015

[9] K. Takasu, T. Saito, T. Yamada and T. Ishikawa, "A Survey of Hardware Features in Modern Browsers", in Proc. of Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS'15), 2015

[10] "Alexa top sites", `https://www.alexa.com/topsites/` (2017.08.28)

[11] "Tor Bug Tracker & Wiki", `https://trac.torproject.org/projects/tor/ticket/6119` (2017.08.28)

[12] "Google Maps Geolocation API", `https://developers.google.com/maps/documentation/geolocation/intro?hl=ja` (2017.08.28)

[13] "Passive OS Fingerprinting: Details and Techniques",

http://www.ouah.org/incosfingerp.htm (2017.08.28)
[14] "Tor Project: Anonymity Online — exit-addresses", https://check.torproject.org/exit-addresses (2017.08.28)

付　　録

表 A·1 識別精度が最良となった組み合わせと先行研究の比較

特徴点	NE	Panopticlick	AmIUnique	本論文
UserAgent 文字列	0.669	✓	✓	✓
画面解像度	0.314	✓	✓	✓
フォントリスト	0.381	✓	✓	✓
タイムゾーン	0.017	✓	✓	
プラグインリスト	0.381	✓	✓	
HTTP Accept Encoding	0.119	✓	✓	
HTTP Accept Language	0.244	✓	✓	
HTTP Connection	0.057		✓	✓
Do Not Track	0.131		✓	✓
Canvas	0.131		✓	✓
Web GL Renderer	0.277		✓	✓
リクエストヘッダ	0.747		✓	✓
platform	0.146		✓	✓
HTTP Accept	0.114	✓		
HTTP クッキーの利用可否	0.001	✓		
SuperCookie	NaN	✓		
WebStorage の利用可否	0.002		✓	
Use of an ad blocker	NaN		✓	
Web GL Vender	0.198		✓	
GPU	0.988			✓
worker	0.983			✓
Web GL	0.492		✓	
タッチの有無	0.113			✓
p0f による推定値	0.926			✓
Tor の IP アドレス	0.021			✓
Google Geolocation API	0.200			✓
ブラウザの種類	0.209			✓
sse	0.220			✓
ssdeep	0.894			✓