5Z-7

HTMLレンダリングエンジンのCSSの挙動差を特徴として用いたブラウザ識別の提案

武居　直樹†　　齋藤　孝道†　　磯　侑斗‡　　桐生　直輝‡
明治大学†　　明治大学大学院‡

1. はじめに

Web 上での行動を追跡することで，利用者の趣味嗜好に合わせた広告を出す，いわゆる，ターゲッティング型広告という仕組みが利用されている．利用者の Web 上の行動を追跡して記録することをトラッキングと呼び，その代表的な手法として，HTTP クッキーを用いる手法がある．その手法ではサーバがブラウザ毎に一意な識別子を割り当てることで利用者を識別して追跡を行う．しかし，利用者により HTTP クッキーが削除された場合や，ブラウザで HTTP クッキーが無効化された場合に，サーバは利用者の識別が不可能となり追跡はできない．

Peter Eckersley [1]により，HTTP クッキーを利用しない手法として，HTTP ヘッダや JavaScript から取得できる特徴を用いて利用者を識別する手法が提案されている．しかし，この手法では JavaScript の実行が不可能な場合に一部の情報が取得できないので，識別できない．

そこで本論文では，ブラウザを通して，サーバ側で利用者を識別するため，利用者が使用しているブラウザやそのバージョンを特徴とする．本論文では，まず，それらの特徴を取得する手法として HTML レンダリングエンジンに CSS を解釈させた際にブラウザ間で挙動が異なることを用いた手法を提案する．また CSS のみを用いた端末の特徴の取得について述べる．

2. Web 上でのトラッキング

この章では Web 上で利用されるトラッキングの手法について説明する．

2.1. HTTP クッキーによるトラッキング

ターゲッティング型広告に利用される手法について説明する．利用者が閲覧した Web ページに広告画像が埋め込まれていた際，その広告画像を提供する広告サーバから HTTP クッキーが発行される．以降，同じ広告サーバの広告画像が埋め込まれた Web ページに利用者がアクセスする度に，広告サーバに HTTP クッキーが送信される．その HTTP クッキーに格納された識別子と利用者が閲覧したページの URL を結び付けることで，広告サーバはトラッキングが可能となる．

この手法ではセッション毎に十分な長さを持つ乱数を識別子として用いるため確実に利用者を特定することができる．しかし，この識別子を削除されるなどすると追跡できなくなる．

2.2. Fingerprint によるトラッキング

ブラウザからサーバに送信される HTTP ヘッダや，利用者のブラウザ上で動作させた JavaScript や Flash などの実行結果をサーバに送信させることで採取できるブラウザの特徴の集合を Fingerprint と呼ぶ．それらの情報を特徴としてとらえ利用者を識別する．

この Fingerprint を用いた手法は，利用者がブラウザを変えた場合に追跡が困難になることや，類似した Fingerprint を持つ利用者が存在する場合に一意に識別できなくなる場合がある[2].

3. 関連技術

3.1. CSS

Cascading Style Sheets (CSS)とは，HTML や XML の要素をどのように表示するかを定義する仕様であり，WWW に関する標準化団体 W3C によって定められている．CSS を用いることで，文字の色や大きさ，見出し，背景や，Web ページ全体のレイアウトといった，ブラウザ上の表示をデザインすることができる．

3.2. HTML レンダリングエンジン

HTML レンダリングエンジンとは，HTML 言語を解釈し，文字や画像をブラウザ上に描画する，ブラウザのプログラムである．HTML，JavaScript や CSS を解釈することができる．

主な HTML レンダリングエンジンとして，

Proposal and Implementation of Browser Identification with CSS Rendering Characteristic
†Naoki Takei
‡Yuto Iso ‡Naoki Kiryu
†Takamichi Saito
Meiji University(†), Graduate school of Meiji University(‡)

Internet Explorer が採用している Trident，Firefox が採用している Gecko，アップルが採用している Webkit，Google Chrome が採用している Blink，Opera がバージョン 12 まで採用していた Presto が挙げられる．

4. 提案手法

4.1. 概要

ブラウザに実装されている HTML レンダリングエンジンは， CSS のプロパティやセレクタの解釈においてブラウザ毎に挙動が異なるケースがある．これら挙動差が生じるプロパティやセレクタを CSS に記述することによって，ブラウザの特徴を，ブラウザ経由でサーバ側において採取しブラウザの識別を行う．

CSS では，PC やプリンタといったメディアのタイプや，画面解像度やデバイスの向きといったメディアの状態を指定できる Media Queries[3]を用いることで，端末の画面解像度，ブラウザのサイズやデバイスの向きに応じて，ブラウザに表示する Web ページのデザインやレイアウトを変えることができる．この仕組みを活用することで画面解像度を取得する．

4.2. 実装状況を用いたブラウザの推測

本論文では，値として外部リソースを用いる CSS のプロパティを利用して特徴を採取する（図 1 参照）．あるプロパティに対応しているブラウザはサーバに外部リソースをリクエストするが，対応していないブラウザでは外部リソースをリクエストしない．同じベンダのブラウザにおいてもバージョンによりプロパティの対応の有無が変わることがある．こういったプロパティを複数組み合わせることで，ブラウザやそのバージョンを特定する．

```
1 body {
2       border-image : url (“test1.jpg”) fill 10 / 10% / 10px
        space;
3       cursor : url(test2.jpg) 2 2, auto ;
4       border-image-source : url(test3.jpg); }
```

図 1. CSS ソースコード例

図 1 の 2 行目で，Google Chrome や Opera はこのプロパティに対応しているので test1.jpg をサーバにリクエストするが，Firefox や IE では対応していないので test1.jpg のリクエストはされない．以下同様に，各ブラウザのプロパティの対応の有無によって test2.jpg や test3.jpg へのリクエストの有無が変わる．

表 1 に図 1 の実行結果を示す．プロパティに対応している場合を○で，対応していない場合を×で示す．

表 1. 各プロパティ対応状況

	border-image:url (“test1.jpg”) fill 10 / 10% / 10px space	cursor:url (test2.jpg) 2 2, auto	border-image-source: url(test3.jpg)
Google Chrome	○	○	○
Opera(15 以降)	○	○	○
Opera (12.16 以前)	×	×	×
Firefox(15 以降)	×	○	○
Firefox(14 以前)	×	○	×
IE	×	×	×

4.3. 画面解像度取得について

```
1   @media screen and (device-width: 1366px) and
    (device-height: 768px) {
2    body {
3      background-image:
       url(test.php?size=1366*768); } }
```

図 2. CSS の Media Queries 指定例（抜粋）

図 2 の 1 行目のように，Media Queries のメディア特性である height と width を用いて 1 ピクセル単位で画面解像度の高さ及び幅を指定する．このような記述を複数用意する．複数の同様の画面解像度について記述がある場合，HTML レンダリングエンジンは適用条件に一致する記述のみを適用する．すなわち，画面解像度と一致する記述の body 内で指定された外部リソースをリクエストし，サーバ側はそのリクエストに付与されたクエリから画面解像度を取得する．

図 2 のコードをブラウザが解釈した際，端末の画面解像度が幅 1366 ピクセルかつ高さ 768 ピクセルの場合のみ，2 行目及び 3 行目の記述が適用される．外部リソースをリクエストする際に「size=”1366*768”」というクエリ変数がサーバ側に届く．よって，利用者の端末の画面解像度を取得することができる．

5. まとめ

本論文では，CSS を用いたブラウザの推定方法と画面解像度及びブラウザのサイズの取得方法の提案をした．これにより，JavaScript が利用できない場合でも特徴点を収集することが可能となった．

今後はブラウザ推定の精度の向上や，あらたな特徴点の採取方法を考えていきたい．

6. 参考文献

[1]Peter Eckersley，2010，
How Unique Is Your Web Browser?
https://panopticlick.eff.org/browseruniqueness.pdf
[2] 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 年暗号と情報セキュリティシンポジウム予稿集．
[3] Media Queries
W3C Recommendation，19 June 2012，
http://www.w3.org/TR/css3-mediaqueries/