Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.30, NO.4, Aug. 2020 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2020.30.4.631

표준 웹 환경 디바이스 핑거프린트를 활용한 이용자 인증모델 연구*

박 소 희,[1†] 장 진 혁,[2] 최 대 선[3‡]
[1]한국교육학술정보원(전문원), [2,3]공주대학교(대학원생, 교수)

A Study on User Authentication Model Using Device Fingerprint Based on Web Standard*

Sohee Park,[1†] Jinhyeok Jang,[2] Daeseon Choi[3‡]
[1]KERIS(Administration Specialist), [2,3]Kongju National University(Graduate student, Professor)

요 약

편리한 인터넷 이용환경을 조성하기 위해 정부에서는 공공·민간 웹사이트의 플러그인 제거 정책을 추진하고 있다. 일반적으로 금융서비스를 제공하는 금융기관 웹사이트는 전자금융거래 안정성 강화를 위해 이상금융거래 탐지시스템을 운영 중이며 이용자의 정보를 수집 및 분석하기 위해 설치형 소프트웨어를 사용하고 있다. 따라서 플러그인 제거 정책에 따라 별도의 소프트웨어 설치 없이 이용자의 정보를 수집할 수 있는 대체 기술 및 대응정책이 필요하다. 본 연구는 표준 웹 환경에서 사용 가능한 디바이스 핑거프린팅 기술들을 소개하고 다양한 기법 중 선택할 수 있는 가이드라인을 제시한다. 그리고 디바이스 핑거프린트를 활용한 머신러닝 기반 이용자 인증모델을 제안한다. 실제로 크롬과 익스플로러 이용자로부터 디바이스 핑거프린트를 수집하여 머신러닝 알고리즘 기반 Multi-class 인증모델을 생성하였으며 실험 결과, 크롬 기반 모델은 약 85%~89%의 성능을 보였으며 익스플로러 기반 모델은 약 93%~97%의 성능을 보였다.

ABSTRACT

The government is pursuing a policy to remove plug-ins for public and private websites to create a convenient Internet environment for users. In general, financial institution websites that provide financial services, such as banks and credit card companies, operate fraud detection system(FDS) to enhance the stability of electronic financial transactions. At this time, the installation software is used to collect and analyze the user's information. Therefore, there is a need for an alternative technology and policy that can collect user's information without installing software according to the no-plug-in policy. This paper introduces the device fingerprinting that can be used in the standard web environment and suggests a guideline to select from various techniques. We also propose a user authentication model using device fingerprints based on machine learning. In addition, we actually collected device fingerprints from Chrome and Explorer users to create a machine learning algorithm based Multi-class authentication model. As a result, the Chrome-based Authentication model showed about 85%~89% perfotmance, the Explorer-based Authentication model showed about 93%~97% performance.

Keywords: Device Fingerprinting, Browser Fingerprinting, No-Plugin, Authentication, Machine Learning

Received(04. 29. 2020), Modified(06. 30. 2020), Accepted(06. 30. 2020)

* 이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원(No.2016-0-00173, 핀테크 서비스 금융사기 방지를 위한 비대면 본인확인)을 받아 수행된 연구임

† 주저자, soheeee0803@smail.kongju.ac.kr
‡ 교신저자, sunchoi@kongju.ac.kr(Corresponding author)

I. 서 론

최근 인터넷 기술이 발전함에 따라 기존 대면 방식을 통한 업무절차들이 온라인 환경으로 전환되면서 이체·예금조회와 같은 은행 업무, 증명서발급·공문서 제출과 같은 행정서비스를 PC와 모바일을 통해 이용할 수 있게 되었다. 대부분의 웹사이트에서는 이러한 서비스를 제공함과 동시에 이용자 보호 및 보안 강화를 위해 플러그인 기능을 사용해왔다. 플러그인(plug-in)은 기존 브라우저가 제공하지 않는 기능을 사용하기 위해 이용자가 별도로 설치해야 하는 소프트웨어를 말하며 주로 본인 확인, 전자서명, 개인정보보호, 키보드 보안 등 보안 기능을 제공하거나 웹 브라우저에서의 문서 열람, 동영상 재생, 증명서 출력 등 추가 기능을 제공하며 이용자의 편의성을 향상시키기 위해 사용된다. 하지만 웹사이트마다 필요로 하는 기능이 다르므로 웹사이트별로 다른 플러그인을 설치해야 한다는 단점이 있다. 또한, 중복설치로 인한 PC 속도 저하, 브라우저 강제 종료 등의 문제점은 웹사이트 이용에 많은 불편함을 초래한다[1].

따라서 정부에서는 편리한 인터넷 이용환경을 조성하기 위해 공공 웹사이트(2,728개)와 민간 500대 웹사이트(국내 인터넷트래픽의 약 83% 차지)를 대상으로 플러그인 제거 정책(no-plugin)을 추진 중이며 기존의 불필요한 플러그인을 제거하고 웹 표준 기술로 대체하거나 무설치 방식의 대안 서비스를 제공하는 방식을 도입하도록 하고 있다[2]. 2019년 정부의 민간·공공 웹사이트 플러그인 현황 발표에 따르면 17년 말 대비 민간 500대 웹사이트 설치 플러그인은 82% 감소하였으며 공공웹사이트 플러그인은 71.6% 감소 된 것으로 나타났다[3].

하지만 플러그인 제거 정책은 웹사이트를 통해 전자금융거래 서비스를 제공하는 금융회사의 보안정책과 충돌한다. 금융회사는 전자금융거래에 대한 위협이 증가함에 따라 안정성 강화를 위하여 이용자의 단말정보, 환경정보 등을 수집하고 분석하여 정상거래와 의심거래를 탐지하고 이상금융거래를 사전에 차단하는 이상금융거래 탐지시스템을 운영하고 있다. 일반적으로 이용자의 정보는 단말정보 수집 모듈, 보안 프로그램과 같은 설치형 소프트웨어를 통해 수집하기 때문에 정부의 플러그인 제거 정책에 따라 소프트웨어 무설치 환경으로 전환하게 될 시 수집·탐지 가능한 정보가 상대적으로 제한되어 이상금융거래 방지의 실효성이 약화될 수 있다. 따라서 소프트웨어 무설치 환경에서 전자금융거래 이용자의 단말정보 수집·탐지 기능을 유지할 수 있는 대체 기술 및 방법에 관한 연구가 필요하다.

본 연구는 소프트웨어 무설치 환경에서의 이용자 식별 대체 기술로서 이용자의 단말정보를 수집하고 조합하여 식별하는 기술인 디바이스 핑거프린팅(Device Fingerprinting)[4]을 소개하고 표준 웹 환경에서 수집 가능한 디바이스 핑거프린트를 조사·분석을 통해 다양한 기법 중 선택할 수 있는 가이드라인을 제시한다. 그리고 머신러닝 기법을 사용하여 표준 웹 환경에서의 디바이스 핑거프린트 기반 이용자 인증모델을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 이상금융거래 탐지시스템과 디바이스 핑거프린팅에 관해 설명하고 3장에서는 표준 웹 환경에서 수집 가능한 디바이스 핑거프린트를 설명한다. 4장에서는 각 핑거프린트의 효과성을 분석하고 선택 가이드라인을 제시하였으며 디바이스 핑거프린트를 활용한 머신러닝 기반 이용자 인증모델을 제안하고 5장에서는 실제 수집한 데이터를 사용하여 이용자 인증모델을 생성하고 그 결과에 대하여 설명한다. 마지막으로 6장에서 결론을 맺는다.

II. 관련 연구

2.1 이상금융거래 탐지시스템

금융회사들은 비대면 방식의 전자금융거래에 대한 위협이 증가함에 따라 안정성 강화를 위하여 이상금융거래 탐지시스템을 통해 이상금융거래를 사전에 탐지하고 차단함으로써 이용자의 피해를 보호하며 최소화한다. 이상금융거래 탐지시스템은 전자금융거래를 이용하는 이용자의 단말정보, 환경정보 등을 수집하고 수집된 정보를 분석하여 정상·의심거래를 탐지하고 이상금융거래를 차단한다. 또한, 모니터링 및 감사 기능을 통해 각 단계를 시각화하고 종합적인 절차를 통합하여 관리하고 있다[5].

탐지의 정확성을 위해 다양한 유형의 정보를 수집하며 수집 정보는 금융서비스를 이용하는 PC와 스마트폰과 같은 단말의 환경정보인 이용자 매체 환경정보와 다양한 이상금융거래 정보인 사고유형 정보로 나누어진다. 이용자의 매체 환경정보는 수집방법에 따라 수집 가능한 정보가 다르며 소프트웨어(플러그인, 수집프로그램) 설치를 통한 수집방법과 웹 기본

기능을 통한 수집방법으로 나뉜다.

소프트웨어 설치형의 경우 별도의 수집프로그램을 배포하여 이용자의 환경정보를 수집하기 때문에 시리얼 넘버, 물리 MAC 등 고유식별정보를 비교적 수집하기 쉬우며 다양한 정보를 수집할 수 있다. 하지만 이용자의 운영체제, 웹 브라우저에 따라 지원하는 플러그인 종류가 다르기 때문에 수집 가능한 정보가 상이하다는 단점을 가진다. 그에 반해 웹 기본 기능만을 사용하여 이용자의 단말정보를 수집하는 경우 별도의 수집프로그램이 필요하지 않고 이용자의 사용환경과 상관없이 정보를 수집 가능하다는 장점이 있다. Table 1.은 정보수집 방법에 따른 특징을 비교한다.

2.2 디바이스 핑거프린팅

디바이스 핑거프린팅이란 웹사이트를 방문한 이용자의 네트워크, 운영체제, 하드웨어, 브라우저 구성 및 성정 정보 등을 기반으로 장치를 트래킹 하는 방법[4]으로 이용자의 단말로부터 수집한 여러 정보를 조합하여 고유한 디바이스 핑거프린트를 생성하여 이를 통해 이용자를 식별한다. 기존에는 이용자로부터 수집한 정보를 기반으로 특성을 분석하여 마케팅, 광고, 서비스 제공 등의 목적으로 활용해왔다.

이용자의 개인정보를 침해하고 개인정보의 자기통제권을 위협할 수 있지만, 보안 측면에서는 이용자의 디바이스를 식별하고 추적·관리함으로써 전자금융거래상의 이상거래를 사전에 차단하거나 추가적인 인증수단으로 활용하여 거래에 대한 안정성 향상에 기여할 수 있다[6].

Table 1. Comparison of Pros and Cons of User Device Information Collection Methods.

	Pros	Cons
Plugin Based	- Interoperation with existing security programs. - Various user information can be collected according to user system.	- The information that can be collected depends on the plug-ins supported by your system.
Web Based	- collection program not required. - Universally available regardless of user system.	- Difficult to collect unusual information for user identification (Hardware, etc.).

웹 브라우저를 통해 수집된 정보를 활용하는 디바이스 핑거프린팅 기술은 브라우저 핑거프린팅이라고도 말하며, 특히 국내와 달리 해외에서는 이용자 식별 및 추적 등 보안 분야에 디바이스 핑거프린팅을 활용한 연구가 활발히 이루어지고 있다. 일부 연구자들은 대량의 다양한 정보를 수집하고 분석하기 위하여 실제 웹사이트를 구축하였으며 웹 방문자는 디바이스 핑거프린트를 확인할 수 있다.

- https://amiunique.org/
- https://panopticlick.eff.org/
- https://browserleaks.com/

2.2.1 디바이스 핑거프린팅 관련 연구

주로 웹 브라우저를 통해 수집된 이용자의 디바이스 정보, 브라우저 및 환경정보 등을 수집하여 이용자를 식별하고자 하는 연구들이 진행되었다.

웹 기반 서비스에서의 익명화된 이용자를 식별하기 위한 많은 방법 중 특히 브라우저 정보의 차이를 사용하는 방법이 연구되었다. 2주 동안 특정 웹사이트를 접속한 방문자의 브라우저 설정, 디스플레이, 플러그인 정보 등을 수집하여 해시를 통해 이용자별 식별자를 부여한 결과, 총 방문자 1328명 중 96.23%가 고유하게 식별되었다[7]. 그 이후 전자프런티어 재단(EFF, Electronic Frontier Foundation)에서는 실제 환경에서의 디바이스 핑거프린팅의 효과성을 검증하기 위해 웹사이트 구축을 통해 대규모의 데이터를 수집하여 핑거프린트를 생성한 결과, 총 470,161개의 핑거프린트 중 83.6%가 고유한 값을 나타내었다. 더불어 기본적인 브라우저 정보 외에 추가정보를 수집한 결과 94.2%가 고유하게 식별되었으며 해당 연구에서 디바이스 핑거프린팅이라는 용어가 처음으로 사용되었다[4].

또한, Mowery et al.은 Amazon Mechanical Turk를 통해 300명의 User Agent 및 WebGL 정보 등을 수집하고 Canvas API를 통해 디바이스의 운영체제, 브라우저 버전, 그래픽 카드 등 다양한 요소에 따라 2D Context를 렌더링한 Canvas Fingerprint와 3D Graphic을 렌더링한 WebGL Fingerprint를 생성하였다. 그 결과, Canvas

Fingerprint는 전체 300개 중 50개의 이미지가 고유하게 나타났으며 WebGL Fingerprinting은 WebGL 지원 불가인 30개를 제외한 270개 중 50개가 고유하게 나타나는 결과를 확인하였다[8]. Canvas Fingerprinting은 현재 주목받는 디바이스 핑거프린팅 기법 중 하나이다.

2.3 국내·외 디바이스 핑거프린팅 활용 동향

최근 온라인상의 전자거래가 활성화됨에 따라 거래의 안정성에 대한 중요성이 높아지고 있으며 더불어 이용자의 이용 및 결제환경에 대한 편의성에 관한 관심이 높아지고 있다. 이에 따라 편의성을 갖추고자 실제 국내외 보안 솔루션 및 이상금융거래 탐지시스템에 디바이스 핑거프린팅 기술이 적용되고 있다.

국내의 빅데이터 기술 및 플랫폼 전문 기업인 인터리젠은 디바이스 핑거프린팅을 활용하여 이용자 단말정보 수집 시 별도의 프로그램 설치 없이 이상금융거래를 탐지하는 iFDS-af를 개발하였다. 웹 표준을 지원하는 웹 브라우저를 통해 IP정보, 브라우저 정보, 시스템 정보 등 50여 가지의 정보로 단말인증키와 보안키를 생성해 단말 인증과 식별 등을 수행한다. 해당 제품은 국내 PG기업에 구축을 완료하여 신용카드, 계좌이체, 다양한 결제환경 등에 적용하여 서비스 중이다[9]. 또한, 해외의 디지털 아이덴티티 전문업체인 ThreatMetrix는 이용자로부터 수집한 IP, Smart ID, Exact ID, 계정 정보 등의 정보를 결합하여 이용자에게 디지털 식별자(Digital ID)를 부여하고 신뢰도를 활용하여 장치인증 및 사기 탐지시스템을 제공하고 있다. 특히 Smart ID는 기존의 Cookie, Token을 사용하지 않고 디바이스와 브라우저의 속성 등을 수집 및 조합하여 생성한 식별자로 쿠키 삭제나 플래시 차단과 같은 상황에서도 디바이스 식별이 가능하다는 장점이 있다[10].

이처럼 디바이스 핑거프린팅은 보안 목적으로 이용자의 접속이력관리, 추적, 블랙리스트, 디바이스 스코어링 등 다양한 형태로 활용되고 있다.

III. 표준 웹 기반 디바이스 핑거프린트

이용자로부터 수집 가능한 정보는 매우 다양하며 소프트웨어 설치 여부에 따라 수집 정보가 상이하다. 본 연구는 별도의 소프트웨어 설치 없이 표준 웹 환경에서 수집 가능한 정보에 중점을 둔다.

표준 웹 환경에서 수집 가능한 디바이스 핑거프린트는 Table 2.와 같으며 W3C의 웹 표준 HTML5를 기준으로 한다. 주로 JavaScript 실행을 통해 정보를 수집하며 IP, JavaScript Attribute, HTTP Request Header에 대한 속성정보를 수집할 수 있다. 또한, 웹 표준 HTML5 API를 활용하여 Canvas, WebGL, AudioContext, WebRTC, CSS3에 대한 각각의 속성정보를 수집할 수 있으며 디바이스의 프로세스 시간 차이를 이용한 디바이스 핑거프린팅 기술인 CryptoFP 등이 있다. 일반적으로 속성 값은 이용자가 사용 중인 디바이스의 운영체제, 브라우저, 하드웨어와 해당 디바이스의 환경설정 등에 따라 달라질 수 있으며 속성 값의 조합을 통해 고유한 디바이스 핑거프린트를 생성한다. 생성된 디바이스 핑거프린트는 이용자 단말 식별과 인증을 위해 사용된다.

Table 2. Device Fingerprint Based on Web Standard

Device Fingerprint	Collected Information
IP	IP address
JavaScript Attribute	Platform, List of Fonts, List of Plugins, Cookie Enabled, Do Not Track, Flash Enabled, Timezone, Memory, Screen Resolution, Use of Local Storage, Use of Adblock, WebGL Vendor, WebGL Renderer
HTTP Request Header	Accept, Accept-Encoding, Accept-Language, User Agent
Canvas	Hash of Rendering Image
WebGL	Hash of Rendering 3D Image
AudioContext	Hash of Audio Signal
WebRTC	Local IP, Public IP
CSS3	CSS3 Attribute Value (Property, Selector, Filter)
Font Metrics	Dimension of Glyph Rendering
CryptoFP	Time-Based Fingerprinting

3.1 IP Address

IP Address는 인터넷규악주소로 컴퓨터 네트워크에서 장치들이 서로 인식하고 통신하기 위해 사용하는 특수한 번호를 말한다[11]. 일반적으로 인터넷

을 사용하기 위해서는 IP 주소를 할당받아야 하지만 공인 IP가 부족해지면서 공유기, 라우터와 같은 장치를 통해 하나의 공인 IP를 할당받아 NAT 방식을 통해 사설 IP를 부여받아 사용하기도 한다. 이러한 특징과 더불어 IP 주소는 네트워크 환경 및 설정에 따라 변화하기 매우 쉬우므로 이용자를 식별하기 위한 정보로 사용하기에 안정성 측면에서 위험성이 높다. 따라서 IP 블랙리스트, IP 우회 여부, 이용자의 지속적인 접속환경인지 등을 판단하여 이용자의 신뢰도 판단에 부가사항으로 활용하는 것이 좋다.

3.2 JavaScript Attributes

JavaScript는 스크립트 언어로 HTML, CSS와 더불어 웹을 구성하는 요소이며 문제 객체 모델(DOM, Document Object Model)을 통해 DOM 구조에 접근할 수 있다.

DOM은 HTML, XML 문서의 프로그래밍 인터페이스로서 웹 문서 내의 요소를 구조화된 객체 모델로 표현해주며 각각의 요소에 접근하는 방법을 제공하여 스크립트가 웹 페이지 내 요소에 접근할 수 있도록 해준다. 따라서 웹 브라우저에서 JavaScript 실행을 통해 환경정보, 설정 정보 등의 이용자 정보를 수집할 수 있다[6,12]. 주로 브라우저 내의 쿠키·추적방지·플래시 설정 여부와 같은 브라우저 설정 값이나 화면 해상도, 시간 등 구성정보에 대한 정보를 수집할 수 있으며[4] Table 3.은 수집 가능한 속성에 대해 나타낸 표이다.

Table 3. JavaScript Attributes

JavaScript Attribute	Detail
Platform	Operating System of Current Browser
Timezone	System time (UTC Offset)
Screen Resolution	Device Screen Resolution (Height, Width, Color Depth)
Cookie Enabled	Whether cookies are used or not.
DoNotTrack	Whether Tracking Prevention is used or not.
Flash Enabled	Whether Flash is used or not.
List of Fonts	List of System Fonts
List of Plugins	List of System Plugins
Use of Local Storage	Whether Local Storage is used or not.
Use of Adblock	Whether Advertising Block Extension is used or not.
Memory	Device Memory (RAM)
WebGL Vendor	Name of Graphic driver vendor
WebGL Renderer	Name of Graphic driver renderer

3.3 HTTP Request Header

HTTP(hypertext Transfer Protocol)는 인터넷에서 클라이언트(이용자의 브라우저)와 웹 서버가 서로 데이터를 전송하기 위한 통신규약으로 클라이언트가 서버로 요청 메시지(Request)를 보내고 서버는 처리한 결과로부터 응답 메시지(Response)로 회신한다. 클라이언트 요청 메시지는 요청문과 헤더 그리고 바디로 구성되어있으며 헤더의 경우 본문 이외의 클라이언트의 브라우저 정보(User Agent), 처리 가능한 파일형식(Accept), 언어(Accept-Language), 인코딩 방식(Accept-Encoding) 등 클라이언트에 대한 정보를 담고 있다[13].

이러한 정보만 사용하여 이용자를 식별하기에는 어려움이 있지만, 클라이언트에 따라 약간씩 다른 값을 갖기 때문에 다른 정보와 조합하여 사용한다면 고유성을 높일 수 있다.

3.4 HTML5 Canvas Fingerprinting

Canvas는 웹 페이지에 그래픽을 그리기 위한 HTML5 API로 주로 데이터 시각화, 사진 조작, 실시간 비디오처리 등 2D 그래픽에 중점적으로 사용된다.

이용자가 Canvas API를 통해 텍스트 또는 이미지를 렌더링할 때 디바이스의 운영체제, 브라우저, 그래픽 카드 등에 따라 렌더링 되는 결과에 차이가 생긴다. Canvas Fingerprinting은 이용자마다 생성된 이미지가 다르다는 특징을 활용하여 동일한 문자나 이미지를 렌더링 하도록 하고 렌더링 된 고유한 이미지에 해시함수를 적용하여 생성된 해시 값을

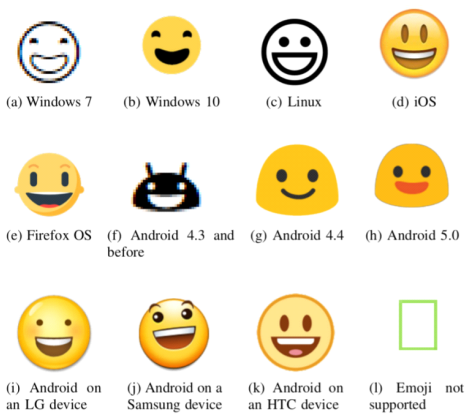

Fig. 1. Rendering Image of U+1F603

식별정보로 사용하는 기술이다[8,14,15]. Fig. 1.은 U+1F603(웃는 얼굴 모양의 유니코드)의 렌더링 결과를 비교한 그림으로 운영체제마다 이미지가 다르고 동일한 운영체제라도 버전에 따라 결과가 다르다는 것을 알 수 있다[16].

3.5 HTML5 WebGL Fingerprinting

WebGL(Web-based Graphics Library)은 HTML5 Canvas API를 통해 3D 구현 시 사용되는 라이브러리로 플러그인 없이 JavaScript를 통해 3D 그래픽 표현이 가능하다.

WebGL Fingerprinting은 Canvas Fingerprinting과 같이 3D 그래픽을 렌더링한 결과의 차이를 이용하는 WebGL Image Hash 기반 핑거프린팅과 WebGL의 지원을 확인하고 User Agent, WebGL Version·Vendor·Renderer·Extensions 등의 정보를 기반으로 하는 브라우저 보고서에 대한 해시 값을 식별정보로 사용하는 WebGL Report Hash 기반 핑거프린팅이 있다[8].

3.6 HTML5 AudioContext Fingerprinting

AudioContext는 Web Audio API로 오디오 모듈과 연동하여 음원을 그래프 화 해주는 Interface이다. AudioContext를 활용한 핑거프린팅은 브라우저를 통해 저주파수의 오디오를 보내고 전송된 오디오 신호의 처리 결과를 식별정보로 사용한다. 이때 마이크, 스피커와 같은 장치 권한 접근이 필요 없으며 이용자 단말의 오디오를 녹음하거나 수집하지 않으며 단말의 오디오 스택만을 기반으로 한다.

AudioContext Fingerprinting은 Oscillator Node에서의 출력신호를 처리하는 방법에 따라 두 가지 방식으로 나뉘며 처리된 최종 출력 값에 해시함수를 적용하여 생성된 해시 값을 식별정보로 사용한다[17]. 이 방법 또한, Canvas Fingerprinting과 유사하게 하드웨어 및 소프트웨어 차이로 처리된 신호의 차이가 발생하는 특징을 활용하며 동일한 조합의 경우 동일한 출력을 생성할 가능성이 있다.

3.7 HTML5 WebRTC Fingerprinting

WebRTC(Web Real-Time Communications)는 웹 브라우저 간의 플러그인 도움 없이 서로 통신이 가능하게 해주는 API로 파일 공유, 음성 및 화상통화 등에 활용된다. 대부분 주요 브라우저에 지원하고 있으나 2020년 2월 기준 Internet Explorer의 가장 최신 버전인 Internet Explorer 11에서는 지원되지 않고 있다.

다른 이용자와의 통신을 위해 WebRTC는 Peer Connection 기능을 사용하여 IP에 대한 정보를 수집하며 이 과정을 통해 이용자의 공인·사설 IP 주소를 획득할 수 있다. 3.1.1에서 언급했다시피 이용자의 IP 주소는 블랙리스트, 우회 여부 등으로 활용할 수 있다[18].

3.8 Font Metrics - based Fingerprinting

Font Metrics는 Font에 따른 Glyph의 크기에 대한 값으로 높이·너비·여백 등과 같은 정보를 반환한다. 이때 동일한 문자라도 폰트, 브라우저, 운영체제에 따라 서로 다른 경계상자(Dimension)로 렌더링 된다. 이러한 특징을 활용하여 경계상자의 높이와 너비 값을 디바이스 핑거프린트로 사용하는 것이 Font Metric-based Fingerprinting 방법이다[19]. 기본으로 지정한 폰트를 통해 동일한 문자를 렌더링한 Metrics 정보를 사용하거나 특정 폰트 목록을 통해 동일한 문자에 대해 렌더링 된 모든 Metrics 정보를 사용할 수 있다.

Fig. 2.는 브라우저에 따른 U+00C6을 렌더링한 결과를 비교하는 그림으로 상단은 Firefox24, 하단은 Chromium 35버전의 브라우저 환경에서 default, sans-serif, serif, monospace, cursiv

Fig. 2. Rendering Results of U+00C6 by Browser (Upper : Firefox24, Lower : Chromium35)

e, fantasy 폰트에 따른 결과를 나타낸다. 그림과 같이 폰트, 브라우저마다 다른 결과가 나타나는 이유는 폰트마다 각 글자를 그리는 경계상자가 정의되어 있으며 이 값이 브라우저마다 상대적인 값을 가지기 때문이다. 따라서 폰트 크기를 지정하더라도 실제 브라우저에 렌더링 되는 경계상자의 크기가 달라진다.

3.9 CSS Attribute Fingerprinting

CSS(Cascading Style Sheets)는 HTML. XML과 같은 웹 페이지의 스타일 요소를 위한 프로그래밍 언어로 배경색, 폰트, 레이아웃 등을 지정하여 문서의 스타일을 꾸며준다. CSS3는 CSS의 가장 최신 버전으로 주요 브라우저에서 대부분 기능을 지원하지만, 일부 기능은 브라우저 종류·버전에 따라 지원되지 않는다.

CSS3 Attribute의 조합을 통해 브라우저를 식별하는 방법을 CSS3 Fingerprinting이라 하며 Session Hijacking 탐지를 위해 처음 제안되었다[20]. CSS3의 Properties, Selectors, Filter의 속성 값을 조합을 통해 이용자의 디바이스 핑거프린트를 생성하는 방법으로 이용자의 실시간 정보와 기존에 저장된 값을 비교함으로써 이용자를 식별한다.

3.10 CryptoFP : Time-based Device Fingerprinting

CryptoFP는 이용자의 디바이스마다 특정 작업을 수행하는데 걸리는 시간의 차이가 존재하는 특징을 활용한 디바이스 핑거프린팅 방법으로 시간차를 이용하기 때문에 Time-based Device Fingerprinting이라고도 한다[21]. 함수를 실행하고 결과까지 걸리는 시간 값을 저장하여 시간 결과행렬(Timing-Matrix)을 생성하고 핑거프린트로 사용하며 시간결과행렬을 생성하는 Generation 과정과 기존 행렬과의 비교를 통해 이용자인지 판단하는 Comparison 과정으로 나뉜다. 특정 작업은 고정되어 있지 않으며 개발자에 따라 달라질 수 있다.

IV. 디바이스 핑거프린트의 효과성 분석

앞서 소개한 바와 같이 표준 웹 기능을 사용하여 이용자로부터 수집 가능한 디바이스 핑거프린트는 매우 다양하다. 이용자를 식별하기 위해 많은 속성의 정보를 사용할수록 식별성(고유성)은 증가할 수 있지만, 정보의 무분별한 수집은 지양되어야 한다. 또한, 디바이스 핑거프린트가 이용자를 특정할 수 없을 만큼 미미한 차이를 가진다면 굳이 사용할 필요가 없으며, 이용자가 주로 사용하는 브라우저에서 디바이스 핑거프린트 수집을 위한 기능을 지원하지 않는다면 수집할 수 없는 문제가 생길 수 있다. 따라서 디바이스 핑거프린트가 충분히 의미 있는 정보인지, 브라우저에서 수집할 수 있는지 등 다양한 기준에 따른 분석이 필요하며 사용 목적에 따라 핵심적인 정보를 선택하여 사용할 필요가 있다.

본 논문에서는 식별성, 브라우저 호환성, 불변성, 오버헤드, 프라이버시를 기준으로 각 디바이스 핑거프린트를 분석하고 분석한 결과를 통해 디바이스 핑거프린트 선택 가이드라인을 제시한다.

4.1 식별성

특정 속성을 가지는 핑거프린트가 얼마의 확률로 고유한지 나타내는 것을 식별성이라고 한다. 식별성은 정보 엔트로피(Entropy)를 측정하여 나타낸다. 엔트로피는 평균 정보량을 말하며 특정 사건이 일어날 확률이 낮을수록 불확실성은 커지고 엔트로피, 즉 정보량은 커지기 때문에 엔트로피가 클수록 핑거프린트가 중복될 확률은 낮아지고 식별성은 높다는 것을 의미한다. 엔트로피는 식(1)과 같이 계산되며 만약 10bit의 정보량을 가진다면 1/1024의 확률로 중복된다고 말할 수 있다.

$$Entropy = -\log_2 pr(X=x) = -\log_2 pr(\frac{x}{X}) \quad (1)$$

Table 4.는 디바이스 핑거프린트에 대한 엔트로

Table 4. Entropy of Device Fingerprinting

DF	Entropy	DF	Entropy
HTTP Request Header	6 - 13	JS_ Screen Resolution	4.5 - 5.5
HTTP_ Accept	0.5 - 1.5	JS_ Cookie Enabled	0 - 0.5
HTTP_ Encoding	0 - 2.0	JS_ DoNot Track	0.5 - 2
HTTP_ Langugae	2.5 - 6.0	JS_ Flash Enabled	0.5 - 1
HTTP_ UserAgent	7 - 10	JS_ List of Fonts	4.5 - 14
Canvas	7.5 - 19	JS_ List of Plugins	5.5 - 15.5
WebGL	8.5 - 9	JS_ Use of Local Storage	0 - 0.5
AudioContext	5 - 5.5	JS_ Use of Adblock	0 - 1
FontMetrics	4.5 - 8	JS_ Memory	2 - 2.5
JS_ Platform	1 - 2.5	JS_ WebGL Vendor	2 - 2.5
JS_ Timezone	0 - 7.5	JS_ WebGL Renderer	3 - 6

피 값을 나타낸 표로 기존 국내외 연구 자료의 검증 결과를 참고하였다[4,6,16,17,19,22,23]. 모든 디바이스 핑거프린트에 대한 엔트로피를 측정하기에는 충분한 데이터와 시간이 필요하며, 따라서 기존 연구에서 엔트로피 결과를 바탕으로 최소-최대범위를 계산하여 Range로 나타냈다. 최소범위와 최대범위는 기존 검증결과의 0.5단위로 내림과 올림 한 값이며 엔트로피는 수집한 데이터 크기, 형식에 따라 달라질 수 있으므로 절대적인 값은 아니다.

4.2 호환성

이용자의 정보는 웹사이트 접속 시 브라우저를 통해 수집하기 때문에 브라우저에서 지원하는 경우 특정 정보는 수집할 수 없는 경우가 발생한다. 또한, 이용자마다 주로 사용하는 브라우저가 다르기 때문에 해당 브라우저에서 디바이스 핑거프린트가 수집 가능한지에 대한 검증이 필요하다. 따라서 국내에서 주로 사용하는 브라우저인 Internet Explorer(IE), Chrome(CH), Firefox(FF), Safari(SF), Opera(OP), Naver Whale(WH)에 대해 디바이스 핑거프린트의 호환성을 조사하였다[24,25].

Table 5.는 브라우저의 최소 버전을 나타내며 JavaScript의 경우 표준 ECMscript5 1.8.5를 기준으로 조사한 호환 버전을 나타낸다. CSS3 Fingerprinting의 경우 속성 조합을 통해 핑거프린트를 생성하며 사용하고자 하는 속성에 따라 지원 브라우저 및 버전이 상이하며 속성에 따른 호환성은 www.caniuse.com 및 www.css3maker.com을 통해 확인가능하다. Naver Whale은 Google Chrome의 오픈소스 엔진인 크로미움 기반 브라우저로 Chrome에서 지원 시 Whale에서도 지원된다.

Table 5. Browser compatibility of Device Fingerprinting

Fingerprinting	Browser				
	IE	CH	FF	OP	SF
IP	O	O	O	O	O
JavaScript Attribute	10+	23+	21+	15+	6+
HTTP Request Header	O	O	O	O	O
Canvas Fingerprinting	9+	4+	36+	10+	4+
WebGL Fingerprinting	11+	8+	4+	121+	5.1+
AudioContext Fingerprinting	X	10+	25+	15+	6+
WebRTC	X	28+	22+	18+	11+
FontMetrics Fingerprinting	O	O	O	O	O
CSS3 Fingerprinting	-	-	-	-	-
CryptoFP	11*	37+	34+	24+	7.1+

*This feature is partially supported.

4.3 불변성

디바이스 핑거프린트의 불변성이란 동일한 이용자에 대해 핑거프린트 값이 얼마나 일정하게 나타나는지를 말한다. 시스템 업데이트, 이용자의 상황 등 환경이 변하게 되면 디바이스 핑거프린트에 영향을 주기 때문에 변화 요인 및 변화 정도에 따른 분석이 필요하다. 디바이스 핑거프린트의 변화 요인은 Hardware의 교체, Software의 업그레이드, 이용자의 상황 변화 그리고 특정 환경설정 변경으로 크게 네 가지로 나눌 수 있으며 Table 6.은 디바이스 핑거프린트의 변화 요인에 따라 변화 정도를 Stable-Less Stable-Unstable로 등급화 하여 나타내었다. 이는 핑거프린트의 변화에 영향을 주는 주요 원인과 실제 데이터를 수집하고 시간이 지남에 따라 속성 변화를 연구한 기존 논문[24,26]을 참고하여 정리하였으며 환경 및 이용자에 따라 다를 수 있다.

Table 6. Browser compatibility of Device Fingerprinting

Status	Fingerprinting	Attribute
Stable	AudioContext Fingerprinting	Hardware (Audio)
	WebRTC	Location (Almost home, company)
	CSS3 Fingerprinting	Configuration
	CryptoFP	Hardware (Graphic card)
Less Stable	IP	Location (Almost home, company)
	Canvas Fingerprinting	Hardware, OS, Browser (Graphic driver)
	WebGL Fingerprinting	Hardware, OS, Browser (Graphic driver)
	FontMetrics Fingerprinting	OS, Browser
Unstable	JavaScript Attribute	OS, Browser
	HTTP Request Header	Browser

4.4 오버헤드

웹사이트 이용자 식별을 위한 정보수집 및 디바이스 핑거프린팅 과정 시 해당 프로세스의 오버헤드로 인해 이용자의 브라우저 이용에 불편함을 제공하지 않도록 해야 한다. 각 디바이스 핑거프린트 추출 과정에 대한 결과속도를 분석한 결과, 주로 milliseconds 내에 수집할 수 있었으며 실제 디바이스 핑거프린팅을 활용한 시스템을 구현할 때에도 최소한의 코드를 이용하여 속도 저하를 최소화하는 것이 필요하다[6,24].

4.5 프라이버시

이용자로부터 수집되는 정보는 프라이버시 침해를 최소화하여야 한다. 디바이스 핑거프린트의 프라이버시를 고려하기 위해 이용자를 식별할 수 있는 개인정보·고유 식별자 및 민감정보 등의 포함 여부를 분석한 결과, 대부분은 침해요소가 존재하지 않았지만, IP의 경우 해당 속성만을 통해 이용자의 디바이스를 특정할 수 있는 고유한 식별자이기 때문에 프라이버시 침해요소가 된다고 볼 수 있다. 이용자의 정보를 수집할 때에는 개인정보 수집 및 이용 동의를 받은 항목에 한하여 수집하여야 하며 최소한의 정보를 수집·이용되어야 한다. 또한, 수집되는 정보는 암호화를 통해 이용자의 정보를 보호할 필요가 있다.

4.6 디바이스 핑거프린트 선택 가이드라인

디바이스 핑거프린트의 식별성, 브라우저 호환성, 불변성, 오버헤드, 프라이버시와 같은 다양한 기준이 존재하며 사용 목적 및 환경에 맞는 기준에 따라 디바이스 핑거프린트의 특성을 고려하여 활용해야 한다. 식별성을 높이기 위해서는 엔트로피가 높은 디바이스 핑거프린트를 중심으로 사용할 수 있으며 이용자들의 특성상 다양한 브라우저를 사용한다면 호환성이 높은 디바이스 핑거프린트를 중심으로 사용하여야 한다.

예를 들어, 식별성을 우선순위로 한다면 엔트로피가 높은 디바이스 핑거프린트를 선택하여 사용한다. 엔트로피의 평균이 5 이상을 가지는 Canvas Fingerprinting, HTTP Request Header, Screen Resolution, List of Fonts, Font Metrics Fingerprinting, WebGL Fingerprinting, HTT

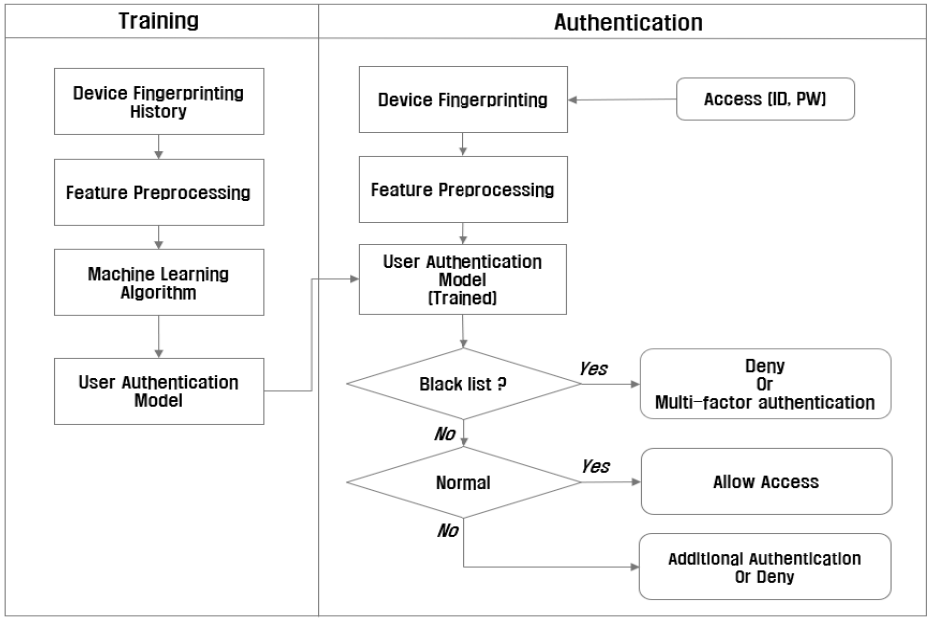

Fig. 3. User Authentication Model Based on Device Fingerprinting Using Machine Learning

P User Agent, AudioContext Fingerprinting 9개가 구성되며 최소 53bit의 식별성을 가지게 된다. 이 경우 AudioContext는 IE가 호환되지 않고, 일부 값은 변화할 가능성이 존재하지만, 프라이버시는 모두 만족하게 된다. 이처럼 각 디바이스 핑거프린트의 특성을 고려하여 구성함으로써 더욱 효과적으로 활용하는 것이 필요하다.

V. 표준 웹 환경 디바이스 핑거프린트 기반 이용자 인증 모델

최근 인공지능 기술이 금융, 의료, 보안 등 다양한 분야에 적용되고 있으며 특히 침입 탐지, 이상 거래탐지, 악성 코드 등과 같은 보안 기술에 인공지능의 머신러닝 기술을 활용한 연구가 많이 등장하고 있다. 이에 따라 본 논문에서는 머신러닝 방법을 적용하여 표준 웹 환경 디바이스 핑거프린트 기반 이용자 인증 모델을 제시한다. 그리고 Javascript를 사용하여 실제 이용자의 크롬과 익스플로러 환경에서의 디바이스 핑거프린트를 수집하고 머신러닝을 사용하여 이용자 인증 모델을 생성하고 성능을 평가하였다.

5.1 머신러닝을 활용한 이용자 인증모델

Fig. 3.은 머신러닝 알고리즘 기반 이용자 모델 생성 및 인증과정을 나타낸다. 이용자의 기존 디바이스 핑거프린트를 생성 후 그 값을 기반으로 머신러닝 알고리즘을 학습하여 인증모델을 생성한 후 이용자가 특정 사이트에 로그인을 시도할 시 현재 디바이스 핑거프린트를 기반으로 학습 된 모델에 이용자의 디바이스 핑거프린트를 입력하여 이용자가 맞는지 판단하며 결과에 따라 접근 허용 및 추가적인 인증과정을 거치게 된다. 머신러닝 기법을 사용하는 경우 충분한 학습 과정을 거쳐야 하며 모델의 목적에 따라 학습 데이터 구성 및 방법이 달라질 수 있다.

예를 들어 정당한 디바이스나 블랙리스트 등 하나의 클래스만을 학습한 모델을 통해 정당한 디바이스, 블랙리스트 인지 아닌지 판단할 수도 있으며 정당한 디바이스, 블랙리스트, 위험도가 높은 디바이스 등 다수의 클래스만을 학습하여 어느 클래스에 해당하는지 판단할 수 있다. 또는 전체 이용자의 디바이스 핑거프린트를 학습하여 어느 이용자인지 판단하는 모델을 생성할 수 있다.

5.2 실험 및 검증

실제 이용자의 디바이스 핑거프린트 특성 및 이를 활용한 이용자 인증모델의 유효성을 검증하기 위해 실제 표준 웹 환경에서의 이용자의 디바이스 핑거프린트를 수집하고 머신러닝 기반 모델을 생성하였다. 수집된 전체 이용자의 디바이스 핑거프린트를 학습하여 특정 이용자의 데이터가 입력되었을 때 어느 이용자인지 판단하는 multi-class classification이다. 본 실험의 경우, 블랙리스트에 속하는 악의적인 이용자의 디바이스 핑거프린트의 특징을 가지는 데이터가 없고 생성해낼 수 없었기 때문에 정당한 이용자만의 데이터를 학습하여 모델을 구성하였다.

5.2.1 데이터 수집

데이터는 전자금융거래를 주요 사용하는 연령대를 고려하여 10대~30대 32명을 대상으로 이용자의 데스크톱과 노트북을 통해 크롬과 익스플로러의 디바이스 핑거프린트를 수집하였다. 수집기간은 약 10일로 이틀 간격으로 한 번씩 1명당 5개의 디바이스 핑거프린트를 수집하였으며 데이터는 크롬 기반 35명의 핑거프린트와 익스플로러 기반 33명의 디바이스 핑거프린트로 구성된다. Fig. 4.는 수집된 이용자의 연령대 및 성별 분포를 나타낸다.

한국인터넷진흥원의 국내 브라우저 이용률(www.koreahtml5.kr, 2019.08)을 기준으로 국내에서 이용률이 가장 높은 2개의 웹 브라우저인 크롬과 익스플로러를 대상으로 하였으며 디바이스 핑거프린트의 특성 상 하드웨어, 운영체제와 더불어 브라우저에 따라 속성 값이 달라지기 때문에 브라우저를 구분하여 데이터를 수집하였다.

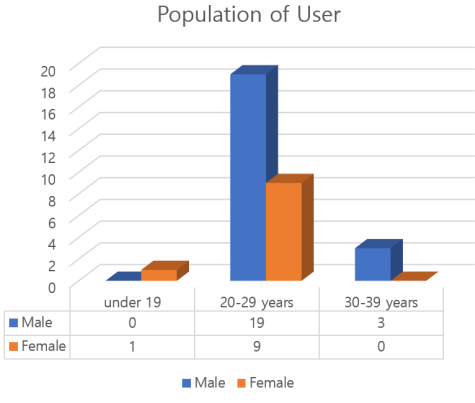

Fig. 4. Population by Age and Sex

이용자들의 PC에서 디바이스 핑거프린트를 수집하기 위해 JavaScript 구현을 통해 데이터 수집 html을 배포하였으며 실제 환경에서는 이용자가 웹 사이트 접속 시 웹 서버에서 데이터를 수집하는 스크립트를 실행하기 때문에 수집하기 위한 파일을 배포하거나 별도의 설치가 필요하지 않는다. 따라서 이용자는 주로 사용하는 데스크톱과 노트북의 웹 브라우저에서 html 실행을 통해 데이터를 수집 및 저장하였다.

이용자로부터 수집한 정보는 앞서 3.1에서 소개한 디바이스 핑거프린트의 일부이며 Table 7.은 실제 수집한 디바이스 핑거프린트와 나타내는 값을 예시로 보여준다. Canvas Fingerprinting의 경우 Canvas API를 통해 2D 이미지를 렌더링한 후 이

Table 7. Collected Device Fingerprinting from User

Device Fingerprinting	Value
Time	2019-11-04 14:32
Host IP	203.253.XX.XXX
Language	en-US, en, ko
Platform	Win32
Cookies Enabled	Unspecified
DoNotTrack	TRUE
List of Fonts	Chrome PDF Plugin, Chrome PDF Viewer, Native Client
List of Plugins	Arial, Arial Black, Arial Narraow, Bahnschrift, …
User Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML like Gecko) Chrome/78.0.3904.87 Safari/537.36
Screen Resolution	1920, 1080, 24
Canvas Fingerprinting (Hash)	1627962531
WebGL Vendor	Google Inc.
WebGL Renderer	ANGLE (NVIDIA GeForce GTX 1050 Direct3D11 vs_5_0 ps_5_0)

미지에 대하여 해시를 하여 해시 값만을 수집하였기 때문에 값이 임의의 숫자를 가진다.

5.2.2 데이터 전처리

이용자로부터 데이터를 수집한 결과, 브라우저에 대하여 동일한 디바이스 핑거프린트 속성을 수집할지라도 브라우저마다 정의한 속성 값 형태가 다르고 반환 값이 다르다, 따라서 크롬, 익스플로러 브라우저별로 데이터를 전처리하고 두 가지의 모델을 생성하였다. 데이터 전처리는 (1) 데이터 제거, (2) 범주형 데이터 변환 과정을 수행하였다.

수집 정보 중 Time과 Host IP는 사용하지 않았다. Time은 디바이스 핑거프린트 수집 시간이고 3.1.1에서 언급했다시피 IP는 이용자의 접속장소에 따라 매우 달라지는 값으로 본 실험에서는 제외하였다. 그리고 Platform, List of Plugins, ColorDepth의 경우 브라우저별 전체 사용자의 값이 모두 동일하므로 제외하였다.

그리고 Language, DoNotTrack, Cookies Enabled, List of Fonts, User Agent, WebGL Vendor, WebGL Renderer는 문자열로 수집된 범주형 데이터이며 수치형 데이터로 변환하기 위해서 One-hot Encoder를 사용하였으며 이 중 Fonts의 경우 전체 사용자로부터 약 250개~300개가 수집되었으나 중복되는 폰트가 약 80개로 이는 제외하여 적용하였다. 그리고 Screen Resolution, Canvas Fingerprinting은 값의 크기가 의미가 없다는 데이터 특성 상 해당 변수들은 모두 One-hot Encoder를 사용하여 0, 1로 이루어진 더미형 변수를 만들었다.

5.2.3 실험 방법

전체 이용자 중 어떤 이용자인지를 판단하는 모델을 생성하기 위해 브라우저별로 모든 이용자를 예측하는 Multi-class 인증모델을 생성한다. 따라서 브라우저마다 모델이 1개씩 생성되며 수집 된 브라우저 환경이 크롬과 익스플로러 2개이므로 총 2개의 식별모델이 만들어진다.

이용자로부터 총 일주일간 수집된 데이터 중 1~4일의 데이터는 학습 데이터로 사용하였으며 5일의 데이터는 테스트 데이터로 사용하여 한 사람당 학습 데이터는 4개, 테스트 데이터는 1개이다. 크롬의 경우 35명, 익스플로러의 경우 33명의 데이터를 기반으로 모델을 학습하였으며 크롬과 익스플로러가 반환하는 데이터의 특성이 서로 다르기 때문에 브라우저에 따라 모델을 생성하였다. 예측에 사용된 머신러닝 알고리즘은 Support Vector Machine, K-Nearest Neighbor, Decision Tree, Random Forest와 Multi-layer Perceptron을 사용하였다.

5.2.4 머신러닝 기반 이용자 인증모델 성능 평가

브라우저별 모델의 성능을 평가하기 위해 혼동행렬(Confusion Matrix)과 정확도(Accuracy)를 측정하였으며 각 클래스의 False Positive Rate(FPR), False Negative Rate(FNR)를 구하여 비교하였다.

혼동행렬은 실제 값과 모델의 예측값이 일치하는지 나타내는 표로 True는 실제 값과 예측값이 일치, False는 실제 값과 예측값이 일치하지 않는 경우를 말하며 Table 8.과 같다[27].

정확도는 전체 데이터 중 예측값이 실제 값과 같은 비율을 말하며 식 (4)와 같이 계산한다. False Positive Rate는 실제 정상(Negative)을 비정상(Positive)이라고 잘못 예측한 비율을 말하며 False Negative는 실제 비정상(Positive)을 정상(Negative)이라고 잘못 예측한 비율을 말한다. 각각 식 (5), (6)과 같이 계산한다. 본 실험은 Multi-class classification으로 각 클래스에 따라 False Positive Rate와 False Negative Rate를 구하고 평균을 계산하여 성능을 비교하였다.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{4}$$

$$False\ \ Positive\ \ Rate = \frac{FP}{FP + TN} \tag{5}$$

$$False\ \ Negative\ \ Rate = \frac{FN}{TP + FN} \tag{6}$$

Table 8. Confusion Matrix

Actual	Predicted	
	Positive	Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

Table 9. Evaluation Result of Multi-class User Authentication Model

Model	Chrome			Internet Explorer		
	Accuracy	FPR (Average)	FNR (Average)	Accuracy	FPR (Average)	FNR (Average)
Support Vector Machine	0.857	0.004	0.143	0.939	0.002	0.061
K-Nearest Neighbor	0.857	0.004	0.143	0.939	0.002	0.061
Decision Tree	0.886	0.003	0.114	0.939	0.002	0.061
Random Forest	0.886	0.003	0.114	0.970	0.001	0.031
Multi-layer Perceptron	0.857	0.003	0.114	0.970	0.001	0.030

모델 생성 후 테스트 데이터를 통해 성능을 평가한 결과, 크롬 기반 인증모델은 Decision Tree와 Random Forest의 정확도가 약 88.6%, Support Vector Machine, K-Nearest Neighbor, Multi-layer Perceptron이 약 85.7%를 보였다. 익스플로러 기반 인증모델은 Random Forest와 Multi-layer Perceptron은 약 97%, Support Vector Machine, K-Nearest Neighbor, Decision Tree가 93.9%를 보였다. 브라우저별 Multi-class 인증모델의 성능 결과는 Table 9.에 나타나 있다.

5.2.5 이전 연구와의 비교

E. Flood 등[28]은 사용자 추적을 막기 위한 방안으로 브라우저 핑거프린팅을 사용해 인증 모델을 만들었다. 이전 연구는 K-Nearest Neighbor 과 12개의 브라우저 핑거프린팅을 사용했으며 Table 10.은 실험에 사용된 디바이스 핑거프린팅을 나타낸다. 본 논문과 사용이 같은 핑거프린팅으로는 Language, Screen Resolution, User-Agent이다.

이전 연구에서의 실험은 모바일 디바이스로 안드로이드, 아이폰, 아이패드를 통해 익스플로러, 크롬 브라우저에서 데이터를 수집하였다. 핑거프린팅을 통하여 사용자를 구분하는 실험의 결과로 익스플로러 86%, 크롬 96%의 성능을 보였다. 이를 통해 본 연구와 상이한 결과를 보이는 주요인으로 디바이스 환경, 사용한 핑거프린팅으로 뽑을 수 있다.

Table 10. Device Fingerprinting used in the experiment

Feature
IP
Language
Screen Resolution
User-Agent
Time Zone
Minimun Clock Error
via Adobe Flash
Round Trip Time
Accept-Language
Accept - Charset

VI. 결 론

본 논문은 웹사이트 노플러그인 환경에서의 전자금융거래 안정성 강화를 위한 대체 기술 및 방안 마련을 위해 표준 웹 환경에서의 사용가능한 디바이스 핑거프린팅 기술을 소개한다. 또한, 표준 웹에서 수집 가능한 디바이스 핑거프린트를 조사하고 각 핑거프린트의 식별성, 호환성, 불변성, 오버헤드 그리고 프라이버시 관점에서의 효과성을 분석함으로써 기준에 따른 선택 가이드라인을 제시하고 디바이스 핑거프린트를 활용한 인증 모델을 제안하였다.

그리고 제안한 방법의 유효성을 보이기 위해 실제 크롬과 익스플로러 이용자의 디바이스 핑거프린트를 수집하여 머신러닝 알고리즘 기반의 Multi-class

인증모델을 생성하였으며 실험 결과, 크롬 기반 인증모델은 최소 85% 이상의 성능을 보였고 익스플로러 기반 인증모델은 93% 이상의 성능 결과를 보였다. 크롬과 익스플로러부터 동일한 정보를 수집하더라고 성능이 다른 결과를 보이는 이유는 브라우저에 따라 속성이 반환하는 값이나 값의 형태가 다르기 때문이며 시간이 지남에 따라 일부 단말 혹은 브라우저의 업데이트로 인해 값이 변했기 때문이다. 실제 이용자에게 추가적인 프로그램 설치를 요구하지 않으면서 이용자의 정보를 수집하여 실험을 통해 디바이스 핑거프린트로 이용자를 식별하기 위한 정보로 사용할 수 있을 것으로 보인다.

본 연구의 데이터 세트는 수집한 디바이스 핑거프린트 중 일부였기 때문에 실제 시스템 상에 디바이스 핑거프린팅을 실질적으로 도입하기 위해서는 다양한 환경과 다양한 디바이스 핑거프린트로 이루어진 대규모의 데이터 세트를 사용해 충분한 연구와 실험으로 효과성 및 안정성을 검증 하는 것이 필요하다.

References

[1] Ministry of the Interior and Safety, "Guidelines for removal of public website plugins," Nov. 2018

[2] Ministry of the Science and ICT, "Guidelines for improving private public website plugins," Nov. 2018

[3] Ministry of Science and ICT, Ministry of the Interior and Safety and Financial Services Commission, "2019 Current status of private and public plug-in improvements," Dec. 2019

[4] P. Eckersley, "How unique is your web browser?," In International Symposium on Privacy Enhancing Technologies Symposium, Springer, Berlin, Heidelberg, pp. 1-19, 2010

[5] "Fraud Detection System Technical guide," Financial Security Agency, Aug. 2014

[6] Seok-eun Jang, Soon-tai Park and Sang-joon Lee, "A Study on Online Fraud and Abusing Detection Technology Using Web-Based Device Fingerprinting," Journal of the Korea Institute of Information Security & Cryptology, Vol. 28, No. 5, pp.1179-1195, Oct. 2018

[7] Mayer, J. R. "Any person... a pamphleteer: Internet Anonymity in the Age of Web 2.0," Undergraduate Senior Thesis, Princeton University, pp.1-103, 2009

[8] K. Mowery and H. Shacham, "Pixel perfect: Fingerprinting canvas in HTML5," Proceedings of W2SP, pp. 1-12, May. 2012

[9] Interezen, http://www.interezen.co.kr/izh6/main/main.jsp, accessed Nov 25, 2019

[10] "Empowering Smarter Authentication and Fraud Decisioning in an Evolving Digital Landscap", LexisNexis, 2019

[11] Wikipedia, "IP address", https://en.wikipedia.org/wiki/IP_address, accessed Nov 25, 2019

[12] TCP school, "Concept of DOM," http://tcpschool.com/javascript/js_dom_concept, accessed Nov 26, 2019

[13] Mozila, "HTTP Header," https://developer.mozilla.org/ko/docs/Web/HTTP/Headers, Mozila Web docs, accessed Nov 26, 2019.

[14] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz, "The Web Never Forgets: Persistent Tracking Mechanisms in the Wild," In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS '14). ACM, New York, NY, USA, pp. 674 - 689, Nov. 2014

[15] P. Laperdrix, G. Avoine, B. Baudry, and N, Nikiforakis, "Morellian Analysis for Browsers: Making Web Authentication Stronger with Canvas Fingerprinting," In International Conference on Detection of Intrusions and Malware and Vulnerability

Assessment, Springer, Cham, pp. 43-66, Jun, 2019
[16] P. Laperdrix, W. Rudametkin, and B. Baudry, "Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints". In 2016 IEEE Symposium on Security and Privacy(SP), IEEE, pp. 878-894, May. 2016
[17] S. Englehardt and A. Narayanan, "Online tracking: A 1-million-site measurement and analysis," In Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, pp. 1388-1401, Oct. 2016
[18] The Spanish Data Protection Agency(AEPD), "Survey on Device Fingerprinting," Feb. 2019
[19] D. Fifield and S. Egelman, "Fingerprinting web users through font metrics," In International Conference on Financial Cryptography and Data Security, Springer, Berlin, Heidelberg, pp. 107-124, Jan. 2015
[20] T. Unger, M. Mulazzani, D. Frühwirt, M. Huber, S. Schrittwieser and E. Weippl, "SHPF : Enhancing http (s) session security with browser fingerprinting," In 2013 International Conference on Availability, Reliability and Security, IEEE, pp. 255-261, Sep. 2013
[21] I. Sanchez-Rola, I. Santos and D. Balzarotti, "Clock around the clock: time-based device fingerprinting," In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, pp. 1502-1514, Jan. 2018
[22] A. Gómez-Boix, P. Laperdrix and B. Baudry, "Hiding in the crowd: an analysis of the effectiveness of browser fingerprinting at large scale," In Proceedings of the 2018 World Wide Web Conference, pp. 309-318, Apr. 2018
[23] P. Laperdrix, N. Bielova, B. Baudry, and G. Avoine, "Browser Fingerprinting : A survey," ACM Transactions on the Web, Vol.14, No.2, Apr. 2020.
[24] F. Alaca and P. C. Van Oorschot, "Device fingerprinting for augmenting web authentication: classification and analysis of methods," In Proceedings of the 32nd Annual Conference on Computer Security Applications, pp. 289-301, Dec. 2016
[25] "Can I Use",https://www.caniuse.com/, accessed Nov 23, 2019.
[26] A. Vastel. P. Laperdrix, W. Rudametkin and R. Rouvoy, "FP-STALKER: Tracking browser fingerprint evolutions," In 2018 IEEE Symposium on Security and Privacy (SP), IEEE, pp. 728-741, May. 2018
[27] S. V. Stehman, "Selecting and interpreting measures of thematic classification accuracy," Remote sensing of Environment Vol, 62, No. 1, pp. 77-89, 1997
[28] E. Flood, J. Karlsson, "Browser Fingerprinting," Master of Science Thesis, Chalmers University, pp.1-99, 2012

〈저자소개〉

박 소 희 (Sohee Park) 학생회원
2018년 2월: 공주대학교응용수학과 학사
2020년 2월: 공주대학교 대학원 융합과학과 석사
2020년 6월: 공주대학교 산학협력단 연구원
2020년 7월~현재: 한국교육학술정보원 전문원
〈관심분야〉 인증, 금융보안, 머신러닝

장 진 혁 (Jinhyeok Jang) 학생회원
2020년 2월: 공주대학교 응용수학과 학사
2020년 3월~현재: 공주대학교 대학원 융학과학과 석사과정
〈관심 분야〉 인증, 금융보안, 머신러닝

최 대 선 (Daeseon Choi) 종신회원
1995년 2월: 동국대학교 컴퓨터 공학과 학사
1997년 2월: 포항공과대학교 컴퓨터공학과 석사
2009년 1월: 한국과학기술원 전산학과 박사
1997년 1월~1999년 6월: 현대정보기술 선임
1999년 7월~2015년 8월: 한국전자통신연구원 인증기술연구실 실장/책임연구원
2015년 9월~현재: 공주대학교 의료정보학과 부교수
2016년~현재: 정보보호학회 이사
〈관심분야〉 인증, 개인정보보호, 이상거래탐지, 의료정보보안, 머신러닝