# 문장 단위 혼합 트래커 대응 도구 설계

신지웅[1], 위성일[2], 전유석[3]
[1]울산과학기술원 컴퓨터공학과 석사과정
[2]울산과학기술원 컴퓨터공학과 교수
[3]울산과학기술원 컴퓨터공학과 교수
zeeshin@unist.ac.kr, seongil.wi@unist.ac.kr, ysjeon@unist.ac.kr

# Mixed Script Guard: Addressing mixed trackers at statement granularity

Zeewung Shin[1], Seongil Wi[2], Yuseok Jeon[3]
[1]Dept. of Computer Science, UNIST
[2]Dept. of Computer Science, UNIST
[3]Dept. of Computer Science, UNIST

요 약

광고 및 트래커 차단기를 사용하는 웹 사용자의 비율이 증가함과 동시에, 트래커 구현 기법 또한 빠르게 발전하고 있다. 최근 트래커 개발자는 다양한 단위 (granularity)에서 웹 사이트의 핵심 기능과 트래커를 묶어 기존 이분법적 분류 및 차단 방식을 제한하고 있다. 정밀한 트래커 탐지 및 디블로팅 필요성이 대두되는 가운데, 트래커의 문장 (statement) 단위 식별과 트래커의 모든 소스 (source)를 고려한 분석 기술의 부재를 확인하였다. 따라서 본 연구는 실행 컨텍스트 (execution context) 기반의 동적 분석 도구 설계를 통해 트래커와 웹 사이트 기능의 리소스 상관관계를 분별함과 동시에, dynamically executed script를 최대한 호출하여 트래커를 문장 단위에서 정밀하게 탐지 및 디블로팅 하는, Mixed Script Guard를 설계하였다.

## 1. 서론

온라인 쇼핑몰, 뉴스, SNS를 비롯한 90% 이상의 웹 사이트는 사용자를 위한 맞춤형 광고와 콘텐츠를 제공하기 위해 트래커에 광범위하게 의존하고 있다 [1]. 트래커는 웹 사이트를 유지하는데 필수적인 요소로, 타 사이트 방문 기록 같은 관심 분야 및 정치적, 종교적 성향 같은 민감한 개인 성향 등의 사용자 데이터를 [1] 수집하여 맞춤형 콘텐츠 경쟁력을 높이고 최고 입찰자에게 해당 데이터를 판매하여 수익을 창출한다 [2].

이를 방지하기 위해 이미 37%의 웹 사용자들이 [3] 트래커 차단기를 (uBlock Origin, Adblock Plus 등) 사용하고 있으며 비율이 증가하는 추세다. 기존 트래커 차단기는 식별된 트래커들을 나열한 filter list (EasyList, EasyPrivacy)를 참고하여 해당 네트워크 엔드포인트를 차단하는 방식을 사용하고 있다. Filter list는 이분법적인 트래커 대응 방식을 채택하는데, 특정 엔드포인트 혹은 리소스에서 트래커가 탐지될 시 차단해도 문제가 없으면 차단 (block), 차단시 웹 컨텐츠, 버튼, 제출 양식 (form), 로그인 기능 등 [4] 사이트 기능에 악영향을 끼칠 시 예외적으로 해당 엔드포인트 혹은 리소스를 허용한다 (exception). 하지만 최근 filter list를 우회하기 위해 웹 사이트의 핵심 기능과 트래커를 같은 네트워크 엔드포인트에서 호출하는 것을 필두로 script, function [5], statement [6] 단위에서 웹 사이트의 핵심 기능과 트래커를 묶는 혼합 트래커 방식이 발견되어 예외적으로 허용하는 케이스가 많아지고 있다. 사용 빈도수의 경우 2021년 기준 크롤링된 script의 6%가 [5], 그리고 2024년 기준 크롤링된 script의 13.4%가 [6] 혼합 트래커로 판별이 되어 점차 늘어나고 있는 중요한 문제임을 알 수 있다.

대다수의 기존 탐지 및 분석 방식은 [7, 8, 9, 10] script 전체를 트래커로 판단하여 혼합 트래커의 유무를 정확하게 판별을 하여도 script를 통째로 차단해 정상적인 웹 사이트 기능또한 비활성화시켜 사용자 경험 (UX)에 치명적인 악영향을 끼치거나 차단을 안 해 유저의 민감한 개인정보를 유출하는 딜레마에 빠지는 한계점이 있다. 최근에 나온 혼합 트래커 탐지 논문 또한 탐지 범위가 함수 단위에서 그쳐 [6] 동일한 함수내 트래커와 웹 사이트 기능이 공존

할 시 문장 단위에선 둘을 분간하지 못하며, 네트워크 리퀘스트 필드 (network request field)만 차단하여 [11] canvas 같은 기존 Web API는 차단하지 못하는 한계점이 있다.

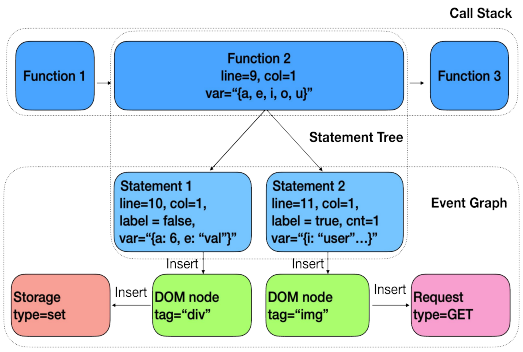
따라서 본 연구는 혼합 트래커의 문장 단위 실행 컨텍스트 (execution context)를 면밀히 분석하여 문장 단위에서 트래커 탐지 및 디블로팅하는 Mixed Script Guard를 제안한다. 효과적인 탐지를 위해서는 트래커와 웹 사이트 기능의 차이를 구별할 수 있는 컨텍스트가 필요하다. 이를 위해 본 연구는 웹 사이트 렌더링 시 호출되는 리소스를 확인할 수 있는 PageGraph [12] 도구를 문장 단위에서 분석 가능하도록 수정한다. PageGraph를 이용한 분석 기법은 가장 면밀하게 리소스 간 상관관계를 확인할 수 있는 동적 분석 환경을 제공한다. 추가로 기존 연구에서 크롤러가 트리거하지 못했던 dynamically executed script를 최대한 호출하기 위해 크롤러에 사용자 행동을 모방한 마우스 움직임, 스크롤 등을 접목하여 초기에 호출된 리소스 외에도 최대한 트래커를 탐지하여 개인정보 보호가 되는 환경 제공을 목표로 한다.

## 2. 이론적 배경

### 2.1 PageGraph

PageGraph는 브라우저에 탑재하는 동적 분석 도구로, 정적 분석 기법과 달리 실제로 사용되고 있는 리소스 간의 관계를 그래프 형태로 보여준다. 기존 PageGraph에 명시된 노드는 페이지 단위에서 사용이 되는 리소스로 DOM node, script, 네트워크 리퀘스트, Web API가 있으며, 엣지는 각 노드 추가, 수정, 삭제등 상관관계를 나타낸다. 해당 방식은 script 위주 코드 기반 정적 분석에 비해 더욱 풍부한 맥락을 제공할 뿐만 아니라, 여러 난독화 기법 (control flow flattening, dead code injection)에 영향을 받지 않는 실제 호출된 관계를 보여줘 더욱 정확하다. 또한 script의 경우 네트워크 리소스로 호출된 script만을 분석하는 정적 분석 기법과는 달리 PageGraph는 inline script 또한 분석할 수 있다.

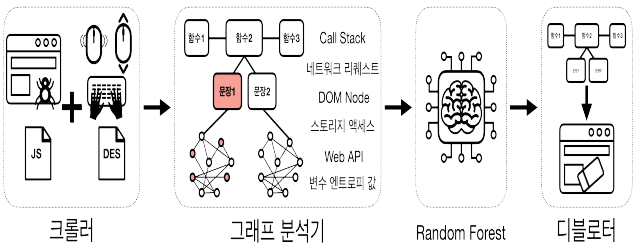
기존 PageGraph의 경우 페이지 단위에서 사용되는 리소스 간 상관관계만 보여줘 문장 단위에서 트래커를 판별할 수 없기에, 본 연구는 PageGraph를 수정하여 [그림 1] 같이 call stack과 statement tree를 추가적으로 명시한다. 또한 기존 event graph 또한 수정하여, script가 아닌 statement가 호출하는 네트워크 리소스의 상관관계를 파악한다. Call stack으로 함수가 호출된 순서를 나타내며, statement tree를 통해 글로벌 변수 값과 함수내 변수 값을 확인하여 문장 단위의 컨텍스트를 보강한다. 기존 PageGraph에서 명시되어 있지 않은 값은 call stack과 statement tree로 보충하여 event graph 내 관련 statement 노드의 metadata 값을 업데이트 해준다.




(그림 1) 수정한 PageGraph의 그래프 구조 예시

### 2.2 Dynamically executed scripts (DES)

Dynamically executed scripts는 웹 사이트 초기 렌더링시 필수적으로 호출되는 script가 아닌 사용자 인풋 및 행동에 따라 호출되는 script를 뜻한다. 해당 script는 보통 DOM node의 event listener 에 명시된 특정 행동조건을 (DOM node에 마우스 올리기, 마우스 스크롤 등) 충족시킬시 호출이 된다. Eval scripts, inline scripts, anonymous function이 tracker function 중 각각 1.5%, 18.3%, 8.2%를 차지하며 [6] 이 중 상당수가 dynamically executed script라는 점을 감안하여 기존 연구의 크롤러를 보강하여 최대한 많은 함수를 트리거하는 것을 목표로 하였다.




(그림 2) Mixed Script Guard 설계 도식

## 3. 디자인

본 연구에서 제안하는 동적 분석 도구인 Mixed Script Guard는 [그림 2] 같이, '크롤러', '그래프 분

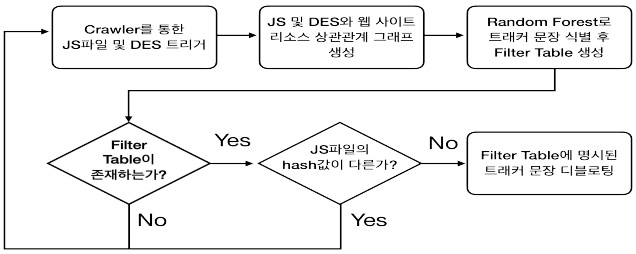
석기', '디블로터'으로 구성되어 있다.

먼저 scalable한 혼합 트래커 판별을 위해 Selenium을 이용하여 Tranco top-10k 웹 사이트의 landing page를 크롤링한다. 이때 몇몇 웹 사이트의 경우 자동 크롤러를 방지하기 때문에 dynamically executed script를 최대한 호출하기 위해 마우스 움직임, 스크롤을 랜덤한 속도 및 방향으로 움직여 우회하고, 입력 유형 박스내 랜덤한 값을 기입하여 유저 활동을 최대한 다양하게 모방한다.

수정된 PageGraph로 생성된 event graph 분석 시 Javascript 문장이 호출한 네트워크 리퀘스트가 filter list에 block 혹은 exception으로 식별이 되어 있을 경우 관련 문장을 true 값으로 soft label하고 트래커 리퀘스트의 숫자를 명시하며, filter list에 등재되어 있지 않은 경우 false 값으로 soft label한다. 이때 exception 처리된 리퀘스트의 경우 혼합 트래커를 포함하기에 block 처리된 리퀘스트와 동일하게 true 값으로 soft label한다.

이후 트래커 관련 네트워크 리퀘스트, DOM node event listener 탑재, 스토리지 액세스, Web API, call stack 등 상관관계와, statement tree에서 얻은 변수 엔트로피 값에 가중치를 두고 random forest 알고리즘으로 기계학습을 하여 트래커 문장을 식별한다. Random forest 알고리즘은 비록 GNN 혹은 DPCNN 방식에 비해 매우 단순한 기계학습 방식이지만, 몇 분내에 트레이닝이 가능할 정도로 빠르고 정확하며 CPU로도 연산이 가능한 가벼운 알고리즘이라 단기간내 웹 사이트 새로고침 혹은 재방문시 트래커를 차단할 수 있다는 장점이 있다. 기계학습 완료 후 Mixed Script Guard 고유의 filter list를 (filter table) 생성하여 웹 사이트마다 트래커로 분류된 문장과 크롤링한 script의 hash 값을 나열하고 해당 값을 바탕으로 디블로팅 할 수 있도록 한다.

마지막으로 filter table에 명시된 트래커로 분류가 된 문장을 line과 column값을 확인하여 body가 비어 있는 문장으로 (empty statement) 치환한다. 이때 트래커 문장에서 사용된 변수를 모두 제거한다면 이후 웹 사이트 기능관련 문장에서 해당 변수를 재호출할 시 undefined 에러로 인해 웹 사이트 기능이 비활성화 될 수 있다. 따라서 call stack과 statement tree에서 해당 변수의 호출 관계도를 확인하여 이후 호출하는 함수 혹은 문장이 기능과 관련 있을 시 변수와 값을 그대로 유지하고, 트래커 함수일 시 값만 비운 상태로 디블로팅한다. 방문한 웹 사이트가 filter table에 명시가 되어 있지 않거나, 크롤링한 script의 hash 값이 명시된 hash 값과 다르다면 event graph를 재생성하여 트래커를 분류한 후 filter table을 업데이트 한다. 웹 사이트의 Javascript 코드가 업데이트 될 시 트래커 구현 방식 또한 달라질 수 있기 때문에 탐지 및 디블로팅 과정을 자동화하여 웹 유저의 개인정보 보호를 최대한 보장한다.




(그림 3) Filter Table 업데이트 설계 도식

## 4. 결론

최근 발견되는 혼합 트래커의 정밀한 탐지 및 디블로팅을 위해서는 문장 단위 실행 컨텍스트 분석과 호출이 가능한 모든 소스를 고려해야 하지만, 기존 연구들에서 이를 모두 만족하는 분석 기술이 필요한 상황이다. 따라서 본 연구는 상기한 문제를 해결하는 Mixed Script Guard를 제안한다. Mixed Script Guard는 문장 단위에서 웹 사이트 리소스 간의 상관관계 및 호출 정보를 확인할 수 있는 동적 분석 환경을 제공한다. 추가로 크롤러에 사용자 행동을 모방한 코드를 접목하여 dynamically executed script를 최대한 트리거하여 관련 트래커 코드를 최대한 탐지 및 디블로팅하여 개인정보 보호가 되는 환경을 제공한다.

## Acknowledgement

이 논문은 2024년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구 결과임 (RS-2024-00437306, 메모리 안전 언어의 적용 확대 및 안전 적용을 위한 통합 플랫폼 기술 개발)과 2024년도 정부(개인정보보호위원회)의 재원으로 한국인터넷진흥원의 지원을 받아 수행된 연구임 (RS-2022-IS000002, 브라우저 상 수집되는 정보주체의 온라인 행태정보 탐지 및 자기 통제기술 개발).

## [참고문헌]

[1] Savino Dambra, Iskander Sanchez-Rola, Leyla Bilge, and Davide Balzarotti. 2022. When Sally Met Trackers: Web Tracking From the Users' Perspective. In 31st USENIX Security Symposium (USENIX Security 22).

[2] Mohammad Ghasemisharif and Jason Polakis. 2023. Read Between the Lines: Detecting Tracking JavaScript with Bytecode Classification. In Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23), November 26 – 30, 2023, Copenhagen, Denmark. ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3576915.3616637

[3] M Malloy, M McNamara, A Cahn, and P Barford. 2016. Ad blockers: Globalprevalence and impact. IMC'16 14-16-Nove (2016), 119 – 125. https://doi.org/10.1145/2987443.2987460.

[4] Alexandra Nisenoff, Arthur Borem, Madison Pickering, Grant Nakanishi, Maya Thumpasery, and Blase Ur. 2023. Defining "broken": user experiences and remediation tactics when ad-blocking or tracking-protection tools break a website's user experience. In Proceedings of the 32nd USENIX Conference on Security Symposium (SEC '23). USENIX Association, USA, Article 203, 3619 – 3636.

[5] Abdul Haddi Amjad, Danial Saleem, Muhammad Ali Gulzar, Zubair Shafiq, and Fareed Zaffar. 2021. TrackerSift: Untangling Mixed Tracking and Functional Web Resources. In Proceedings of the 21st ACM Internet Measurement Conference. Association for Computing Machinery.

[6] Amjad, Abdul Haddi, et al. "Blocking Tracking JavaScript at the Function Granularity." arXiv preprint arXiv:2405.18385(2024).

[7] Umar Iqbal, Peter Snyder, Shitong Zhu, Benjamin Livshits, Zhiyun Qian, and Zubair Shafiq. 2020. Adgraph: A graph-based approach to ad and tracker blocking. In 2020 IEEE Symposium on Security and Privacy (SP). IEEE, 763-776.

[8] Sandra Siby,Umar Iqbal,Steven Englehardt, Zubair Shafiq, and Carmela Troncoso. 2022. WebGraph: Capturing Advertising and Tracking Information Flows for Robust Blocking. In 31st USENIX Security Symposium (USENIX Security 22).

[9] Quan Chen, Peter Snyder, Ben Livshits, and Alexandros Kapravelos. 2021. Detecting Filter List Evasion with Event-Loop-Turn Granularity JavaScript Signatures. In 2021 IEEE Symposium on Security and Privacy (SP).

[10] Changmin Lee and Sooel Son. 2023. AdCPG: Classifying JavaScript Code Property Graphs with Explanations for Ad and Tracker Blocking. In Proceed- ings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23), November 26 – 30, 2023, Copenhagen, Denmark. ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3576915.3623084

[11] Shuang, He, Lianying Zhao, and David Lie. "Dumviri: Detecting Trackers and Mixed Trackers with a Breakage Detector." arXiv preprint arXiv:2402.08031(2024).

[12] Alexander Sjösten, Peter Snyder, Antonio Pastor, Panagiotis Papadopoulos, and Benjamin Livshits. 2020. Filter List Generation for Underserved Regions. In Proceedings of The Web Conference 2020 (WWW '20). Association for Computing Machinery, New York, NY, USA, 1682 – 1692. https://doi.org/10.1145/3366423.3380239