Heinrich C. Mayr, Martin Pinzger (Hrsg.): INFORMATIK 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 1691

Integration von User Privacy Mechanismen in clientseitige Standardbibliotheken

Hauke Coltzau[1]

Abstract: In diesem Beitrag wird überblicksartig diskutiert, wie clientseitige Standardbibliotheken wie jQuery [JQ16] erweitert werden können, um Möglichkeiten zur verdeckten Profilbildung, sowie zum Fingerprinting und Tracking von Nutzern über Webseitengrenzen hinweg einzuschränken. Er dient als Diskussionsgrundlage für nachfolgende Implementierungstätigkeiten.

Keywords: User Privacy, Datenhoheit.

1 Verdeckte Profilerzeugung und User Tracking

Den vielen Vorteilen, die sich aus der Nutzung smarter, mobiler Endgeräte im Alltag ergeben, steht unter anderem die direkte Bedrohung der Privatsphäre und Anonymität ihrer Nutzer vor allem gegenüber Dienst- und Inhaltsanbietern gegenüber. Dies geschieht zum einen durch die als freiwillig anzunehmende Datenherausgabe der Kunden an soziale Onlinenetzwerke, zum anderen aber durch verdeckte Sammlung und Analyse von Nutzerdaten über Betreibergrenzen hinweg und unter Einsatz von Big-Data Methoden.

Canvas Fingerprinting (u.a. [AEE14]) und Nutzerverhaltensanalyse ([ARA11]) zeigen exemplarisch, wie es auf Anbieterseite gelingen kann, ohne Verwendung üblicher Authentifizierungsverfahren über Systemgrenzen hinweg Nutzer mit hoher Sicherheit zu identifizieren. In Kombination mit Realweltdaten, die durch die große Anzahl von Sensoren in modernen Smart Devices problemlos erfasst werden können, entstehen umfangreiche Profile der Nutzer, über deren Erstellung, Auswertung und Weitergabe ebendiese keinerlei Kontrolle ausüben können. Die mittlerweile als permanent anzunehmende Vernetzung der Geräte in Kombination mit deren eigener Leistungsfähigkeit ermöglicht die Erstellung und Übertragung von Profildaten in erheblichem Umfang und mit feiner zeitlicher Granularität. Der Impact ist besonders groß, wenn weitere, nicht kontextzugehörige Daten beispielsweise aus sozialen Netzwerken zur Profilerstellung herangezogen werden.

Datenanalyse im Backend und Verknüpfung von auf den ersten Blick unzusammenhängenden Daten führen zu Erkenntnissen, auf deren Basis ohne tatsächliche Kenntnis der Person unternehmerischen Entscheidungen getroffen werden. Diese wiederum können sich erheblich auf den Nutzer und seinen Alltag auswirken, wie

[1] Fernuniversität Hagen, hauke.coltzau@fernuni-hagen.de

es beispielsweise bei wohnortbezogener Bonitätsbewertung bereits gängige Praxis ist. Im höchsten Maße bedenklich ist hierbei, dass Nutzer weder Kenntnis darüber besitzen, welche Daten genutzt werden, noch Einfluss nehmen können, die Auswertung, Interpretation und Weitergabe derselben zu steuern.

Es liegt also im Interesse der Nutzer, der ungesteuerten und für sie unkontrollierbaren Verwertung eigener Daten sowie der Profilerzeugung und -nachverfolgung entgegenzuwirken. Aufgrund vielfältiger Szenarien und Kontexte, in denen Profildaten entstehen und ausgewertet werden können, scheint ein einheitlicher und umfassender Ansatz, der alle diese Szenarien abdeckt, in der Praxis nur schwer umsetzbar. Stattdessen erscheint es zielführender, den Nutzern eine Menge unabhängiger Werkzeuge und Hilfsmittel zur Verfügung zu stellen, die sie an ihre jeweiligen Bedürfnisse und Anwendungskontexte anpassen können.

2 User Privacy

So naheliegend und fundiert die Kritik an freiwilliger Herausgabe eigener Daten beispielsweise bei Registrierung in und Nutzung von sozialen Onlinenetzwerken (Facebook et. al.) auch ist, so stellt dieses Verhalten aus Sicht des Autors und im Kontext dieses Beitrags keine Verletzung der User-Privacy dar. Schließlich stimmt der User dieser Weitergabe und der daraus resultierenden weitläufigen Nutzung aktiv zu und kann sich in Anbetracht breiter gesellschaftlicher Diskussion auch nicht auf Unwissenheit berufen. Eine technische Lösung ist hier weder zielführend noch ist sie geeignet, an Stelle des notwendigen gesellschaftlichen und politischen Diskurses zu treten.

Im Fokus der diskutierten Fragestellung stehen stattdessen diejenigen Nutzerdaten, für die mindestens drei der folgenden Kriterien zutreffen, wobei die Grenzen zwischen zulässiger und unzulässiger oder fragwürdiger Datenverwertung fließend sind, sodass diese Kriterien selbst nur Richtlinien sein können:

- sie werden ohne Zustimmung der Nutzer erhoben
- ihre Erhebung erfolgt individualisiert
- der Umfang ihrer Erhebung ist unklar
- ihre Erhebung dient nicht oder nicht hauptsächlich dem für den Nutzer erkennbaren Zweck der besuchten Seite
- sie werden ohne explizite Zustimmung über Seiten-/Betreibergrenzen hinweg geteilt
- sie werden in einer Art und Weise verwendet, die geeignet ist, für den Nutzer nicht transparente, an sich vermeidbare Nachteile zu generieren.

Während über die Verwendung der Daten im Backend und deren weiterer Auswertung naturgemäß nur wenig Informationen vorliegen, ist über Techniken des Profilings, des Fingerprintings und des Trackings bekannt, dass diese weitflächig eingesetzt werden.

Beim Profiling (s. [M+14][PHV+13]) werden vom Nutzer explizit oder implizit zur Verfügung gestellte Informationen zur Erstellung eines individuellen (ggf. auszugsweisen) Benutzerprofils herangezogen. Hierbei lässt sich grob an Hand der verwendeten Quellen unterscheiden: Öffentliche Quellen stehen dem Grunde nach ohne nennenswerte Beschränkungen für jedermann zum Profiling zur Verfügung. Hierzu zählen im Wesentlichen soziale Onlinenetzwerke aber auch Onlineangebote von Lokalzeitungen und privaten Betreibern (Vereine, etc.). Diese Form des Profilings kann mit Hilfe von Big-Data Methoden auf eine große Anzahl von Nutzern angewendet werden (s.a. [HRN13]).

Geschlossene Quellen entstehen dort, wo Nutzer in einem als privat angenommenen Verhältnis zu einem Dienstanbieter bei der Inanspruchnahme der Dienste Daten erzeugen, die zum Profiling durch den Dienstanbieter selbst geeignet sind. Diese Daten können hochsensibler und sehr persönlicher Natur sein, wie es beispielsweise bei E-Mails oder Kalenderdaten der Fall ist. Entsprechend „wertvoll“ sind diese Daten für den Dienstanbieter.

Gegen Profiling aus offenen Quellen können Nutzer vor allem durch Nicht-Veröffentlichung entgegenwirken, technische Lösungen stehen hier nicht im Vordergrund. Die Abwehr von Profiling aus geschlossenen Quellen gestaltet sich bei der Nutzung von Onlinediensten ungleich schwieriger, da hierzu in den meisten Fällen ein Mitwirken des Dienstanbieters vorausgesetzt ist, der jedoch das Profiling im eigenen Interesse durchführt. Auch hier sind nicht in erster Linie technische Lösungen gefragt.

Die Nachverfolgung besuchter Seiten und ggf. deren Navigation durch diese Seiten wird als (User-)*Tracking* bezeichnet. Hierzu müssen Nutzer eindeutig identifizierbar sein (nicht notwendigerweise jedoch *persönlich* identifizierbar). Soll diese Identifizierbarkeit auch ohne Wissen und Zutun des Nutzers erfolgen, spricht man von *Fingerprinting*. Klassische Verfahren wie die Verwendung von Cookies sind aufgrund ihrer guten Abwehrbarkeit auf dem Rückzug. Neuere Verfahren (s. u.a. [AJN+13], [NKJ+13]) versuchen, den User über Eigenschaften seines Browsers (Canvas Fingerprinting), seines Systems oder über das Verhalten des Nutzers selbst (bspw. durch Mausbewegungen) zu identifizieren. Anonymes Browsing, wie es beispielsweise von *The Onion Router* (TOR) zur Verfügung gestellt wird, verschleiert nur den Absender, schützt jedoch keinesfalls sicher vor Tracking und Profiling.

Nikiforakis et. al. beschreiben in [NJL15] Varianten zur Verschleierung von für das Fingerprinting geeigneten Daten durch gezielt eingefügte Zufallsinformationen und zeigen dessen Wirksamkeit gegen einige in der Praxis verbreitete Tools im Rahmen einer eigenen Browserimplementierung. Ihre Arbeit kann als Grundlage für diejenigen im nachfolgenden Kapitel diskutierten Erweiterungen genutzt werden, in denen Randomisierung von Daten verwendet werden soll.

Allen Verfahren zum Fingerprinting und Tracking ist gemein, dass sie nicht ohne aktiven Code auf Seite des Clients durchgeführt werden können. Der sicherste Ansatz zur Vermeidung von Fingerprinting ist daher zweifellos, die Ausführung von clientseitigen Skripten zu unterbinden. Dies führt jedoch dazu, dass eine Vielzahl von Webanwendungen und Webseiten nicht oder nur sehr eingeschränkt nutzbar ist. Skriptblocker wie *NoScript* verhindern die Ausführung einzelner Skripte, nichtversierte Nutzer können jedoch kaum fundierte Entscheidungen treffen, welche Skripte zulässig sind und welche ihre Datenhoheit untergraben.

Automatisierte Lösungen zur Vermeidung von Tracking und (in Grenzen) Profiling wie *Privacy Badger*, *Disconnect Private Browsing* und *Ghostery* unterbinden die Ausführung von bekannten Clientbibliotheken mit Trackingfunktionalität. Sie sind jedoch browserabhängig und setzen stetige Pflege im Sinne der Aufnahme neuer Blockadekandidaten voraus.

3 Möglichkeiten zur Datenkontrolle an der Schnittstelle zum User

Eine Alternative zum Blockieren von Skripten kann sein, Standardbibliotheken an der Schnittstelle zum User derart zu härten, dass dem Nutzer erfasste und übertragene Daten transparent gemacht werden und er deren Übermittlung nach seinen Bedürfnissen unterbinden oder steuern kann.

Zur Implementierung solcher Schutzansätze bietet sich eine Erweiterung der jQuery-Bibliothek ([JQ16]) an, weil diese Bibliothek als leistungsfähiges Standardwerkzeug nahezu alle Aspekte der Implementierung von Clientanwendungen abdeckt und einen besonders hohen Verbreitungsgrad hat. Sie bietet Schnittstellen zur Erfassung von Nutzerereignissen, zur Manipulation des DOM und zur asynchronen Kommunikation im Hintergrund an und ist damit eigentlich als Werkzeug zur Aushöhlung der Datenhoheit des Nutzers prädestiniert.

Nachfolgend wird übersichtsartig gezeigt, welche Eingriffsvarianten sich direkt in jQuery implementieren lassen und auf welche bekannten Verfahren zur Vermeidung von Fingerprinting, Tracking und ungewollter Datenerhebung im Allgemeinen dabei zurückgegriffen werden kann.

3.1 Kommunikationsbeschränkung

Eine der Kernfunktionen von jQuery ist die Durchführung synchroner und asychroner http-Requests sowie die Behandlung der entsprechenden Antworten. Asynchrone Requests eignen sich naturgemäß besonders für Datenübertragung im Hintergrund, aber auch über eingeschleuste Parameter synchroner Requests lassen sich am Nutzer vorbei Daten übertragen.

Ähnlich wie in bestehenden Browserplugins kann die Möglichkeit zum Absetzen von http-Requests gezielt eingeschränkt werden. Sowohl Vermeidung von Cross-Site Kommunikation als auch Techniken wie Black- und Whitelisting kommen als naheliegende Varianten in Frage. Durch die weite Verbreitung von jQuery ist eine communitygetriebene Pflege der jeweiligen Listen ein realistischer Ansatz.

Eingeschleuste Parameter lassen sich auf diesem Wege jedoch nur begrenzt erkennen. Dies gilt insbesondere dann, wenn durch Verschlüsselung oder ähnliche Verfahren eine Analyse derselben nicht möglich ist. Als einzige Maßnahme bleibt in diesem Fall die Information des Netzers über potentielle Verletzungen seiner Datenhoheit.

3.2 Content Sandboxing

Bereits bevor Daten versendet werden, kann daher eine Beschränkung des Zugriffs auf diese sinnvoll sein. Inhalte von Bedienelementen wie Suchfeldern sind im Regelfall nur für einen kleineren Teil der Anwendung von Bedeutung und können vor dem Zugriff durch andere Anwendungszweige geschützt werden, ohne dass Funktionsverluste entstehen. Detektionspunkte entstehen im Wesentlichen dort, wo auf DOM-Elemente aus mehreren Skripten heraus zugegriffen wird. Die Abwägung, ob ein Zugriff generell erlaubt wird und welche Informationen je nach aufrufendem Skript zur Verfügung gestellt werden, ist sicher nicht trivial. Dennoch kann das Auftreten von skriptübergreifendem Zugriff zumindest als Indiz für ungewollte Datenerhebung gewertet werden und im einfachsten Fall auch hier zu einer Information des Nutzers führen, selbst wenn keine weiteren Aktionen daraus abgeleitet werden.

3.3 Ereignisnormalisierung und -verschleierung

Eine zentrale Anlaufstelle zur Erfassung von Verhaltensdaten ist die Ereignisverwaltung auf dem Client. Jedes Ereignis, dessen Auslösecharakteristik entweder durch den Nutzer oder durch das verwendete System beeinflusst wird, ist ein potentieller Anknüpfpunkt für Tracking und Profiling. Sobald jedoch sichergestellt wird, dass sämtliche Ereignisse durch eine geeignete Normalisierungs- oder Verschleierungskomponente gefiltert werden, bevor sie zur Bearbeitung an registrierte Listener weitergeleitet werden, sind die Möglichkeiten hierzu wesentlich stärker beschränkt. Dieses Verfahren kann vollständig transparent durchgeführt werden.

Verschleierung und Normalisierung sind unterschiedliche Ansätze, den Informationsgehalt der Ereignisdaten zu reduzieren. Bei der Ereignisverschleierung wird in dem Ereignis in geeignetem Maße Information verändert, was dazu führt, dass mehrere gleichartige Ereignisse weniger voneinander unterscheidbar sind und somit die Ereignischarakteristik nicht mehr oder zumindest nur in reduziertem Maße zum Fingerprinting herangezogen werden kann. Dasselbe Ziel verfolgt die Normalisierungskomponente, die mitgelieferte Ereignisdaten auf ihre wesentliche Aussage reduziert. So ist beispielsweise bei der Behandlung von Mausklick-Ereignissen

im Regelfall nur relevant, welche DOM-Elemente davon betroffen sind, nicht aber die exakten Koordinaten des Klicks. Dennoch müssen Normalisierung und Verschleierung dynamisch an die Bedürfnisse des Nutzers und der jeweiligen Anwendung angepasst werden können, um Funktionalitätseinbußen zu vermeiden.

Selbstverständlich können Skripte, die Daten für Fingerprinting erfassen sollen, eigene Ereignisbehandlungsfunktionen implementieren und diese an jQuery vorbei direkt über die Javascript Kernfunktionen registrieren. Hier lassen sich jedoch die Sprachmöglichkeiten zum sehr einfachen Überladen von Funktionen in Javascript direkt nutzen, um sicherzustellen, dass die diskutierte Erweiterung von jedem dieser Registrierungsversuche Kenntnis erlangt und diese über die eigenen Behandlungsroutinen transparent umlenkt.

3.4 Ereigniskettenanonymisierung

Während Ereignisnormalisierung und -verschleierung nur das einzelne Ereignis betrachtet, richtet sich Ereigniskettenanonymisierung gegen die Ermittlung von Erkenntnissen, die sich aus der Kombination mehrerer in Zusammenhang stehender Ereignissen ableiten lassen, also beispielsweise einer Analyse des Zeitverhaltens bei der Tastatureingabe oder bei Mausbewegungen oder der Verwendung des Scrollrades. Normalisierung und Verschleierung sind bei solchen Ereignisserien leicht durchführbar, beispielsweise durch Einfügen von interpolierten Blindereignissen, durch Auslassung oder Verzögerung von Ereignissen, wenn Folgeereignisse existieren oder durch Glättung von Bewegungskurven.

3.5 Anonyme geteilte Profilnutzung

Noch einen Schritt weiter führt der Gedanke, zusammenhängende Ereignisketten (also Nutzerverhalten) anonymisiert über mehrere Systeme hinweg als Pseudoverhaltensprofile zu teilen und anstelle der vom Nutzer angestoßenen Ereignisketten zu verwenden. Über die Zeit oder mit Wechsel auf andere Webseiten werden auch die verwendeten Profile gewechselt. Fingerprinting ist dann zwar möglich, verfehlt aber seinen Zweck, denn weder wird der eigentliche Nutzer sicher identifiziert noch ist Tracking über mehrere Webseiten hinweg möglich.

3.6 Systemanonymisierung

Fingerprinting auf Basis von Systemeigenschaften (Canvas Fingerprinting und ähnliche Verfahren) macht sich zu Nutze, dass sich Systeme in ihrem Verhalten und in ihrer Konfiguration in vielen Details unterscheiden. Die Analyse einzelner Eigenschaften (wie beispielsweise die Anzahl installierter Schriften) reicht zur Identifikation typischerweise nicht aus, ermöglicht aber durch Kombination eine oft ausreichend hohe Identifikationsgenauigkeit.

Einige der herangezogenen Eigenschaften wie beispielsweise Variationen bei der Darstellung von Texten betreffen noch Grenzbereiche der Zuständigkeit von jQuery, weil die dazu notwendigen Operationen das DOM manipulieren und Informationen aus diesem abfragen. Viele Systemeigenschaften wie Details zur Browserumgebung liegen jedoch klar außerhalb der Kompetenz von jQuery. Es wäre demnach eine eigenständige Erweiterung zu implementieren.

Die Abwehrmöglichkeiten sind dem Verfahren nach ähnlich zu den bereits diskutierten Ansätzen. Systemeigenschaften lassen sich ausblenden oder manipulieren (beispielsweise die Anzahl geladener Plugins), randomisieren oder mit anderen Nutzern über Systemgrenzen hinweg teilen.

Fingerprinting auf Basis der Leistungsfähigkeit des Systems (CPU Rechenzeit für Testaufgaben, verfügbarer Arbeitsspeicher) lässt sich mit diesen Mitteln jedoch nicht oder nur schwer vermeiden. Hier sind systemnähere Ansätze gefordert.

4 Zusammenfassung und Ausblick

In diesem explizit als Ideenpapier zu verstehenden Beitrag werden Ansätze diskutiert, clientseitige Standardbibliotheken wie jQuery derart zu erweitern, dass ungewollte Datenerfassung und –übertragung eingeschränkt werden kann. Die Vorschläge richten sich insbesondere gegen Fingerprintingtools und Tracker und erheben dabei keinen Anspruch auf Vollständigkeit.

In zukünftigen Arbeiten sollen die hier vorgeschlagenen Ansätze im Rahmen eines quelloffenen Projektes implementiert und ihre Wirksamkeit gegenüber bekannten Tracking- und Fingerprintingwerkzeugen überprüft werden.

Literaturverzeichnis

[AEE14] Acar, G. et al.: The web never forgets: Persistent tracking mechanisms in the wild. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2014. S. 674-689.

[ARA11] Angeletou, S.; Rowe, M.; Alani, H.: Modelling and Analysis of User Behaviour in Online Communities. In: (Aroyo, L. et.al. Hrsg.): Proc. 10th International Semantic Web Conference, Bonn 2011. Springer, Berlin/Heidelberg S. 35-50, 2011.

[JQ16] jQuery Projektseite, jquery.com, Stand: 27.06.2016.

[NJL15] Nikiforakis, N.; Joosen, W.; Livshits, B.: PriVaricator: Deceiving Fingerprinters with Little White Lies. Proc. 24th International Conference on World Wide Web. 2015. ACM, New York, S. 820-830.

[M+14] Mitrou, Lilian, et al.: Social media profiling: A Panopticon or Omniopticon tool? In: Proc. of the 6th Conference of the Surveillance Studies Network. 2014.

[PHV+13] Peñas, P.; del Hoyo, R.; Vea-Murguía, J.; González, C.; Mayo, S.: "Collective Knowledge Ontology User Profiling for Twitter -- Automatic User Profiling," Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013, Atlanta, 2013, S. 439-444.

[HRN13] Hoppe, A. ; Roxin, A.; Nicolle, C.: Dynamic, Behavior-Based User Profiling Using Semantic Web Technologies in a Big Data Context", In Proc. On the Move to Meaningful Internet Systems: OTM 2013 Workshops, Graz, 2013, Springer, Berlin/Heidelberg, S. 363-372.

[AJN+13] Acar, G.; Juarez, M.; Nikiforakis, N.; Diaz, C.; Gürses, S.; Piessens, F.; Preneel, B.: FPDetective: dusting the web for fingerprinters. In Proc. ACM SIGSAC conference on Computer & communications security (CCS '13), 2013, ACM, New York, 1129-1140.

[NKJ+13] Nikiforakis, N.; Kapravelos, A.; Joosen, W.; Kruegel, C.; Piessens, F.; Vigna, G.: Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting. In Proc. IEEE Symposium on Security and Privacy (SP), 2013, Berkeley, S. 541-555