



Lehrstuhl für Informatik 1
Friedrich-Alexander-Universität
Erlangen-Nürnberg



MASTERARBEIT

Maßnahmen gegen Nutzerverfolgung mittels Browserfingerprinting

Tim Grocki

Erlangen, 3. Oktober 2014

Examiner: Prof. Dr. Felix Freiling
Advisor: Dr. Zinaida Benenson

Eidesstattliche Erklärung / Statutory Declaration

Hiermit versichere ich eidesstattlich, dass die vorliegende Arbeit von mir selbständig, ohne Hilfe Dritter und ausschließlich unter Verwendung der angegebenen Quellen angefertigt wurde. Alle Stellen, die wörtlich oder sinngemäß aus den Quellen entnommen sind, habe ich als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

I hereby declare formally that I have developed and written the enclosed thesis entirely by myself and have not used sources or means without declaration in the text. Any thoughts or quotations which were inferred from the sources are marked as such. This thesis was not submitted in the same or a substantially similar version to any other authority to achieve an academic grading.

Der Friedrich-Alexander-Universität, vertreten durch den Lehrstuhl für Informatik 1, wird für Zwecke der Forschung und Lehre ein einfaches, kostenloses, zeitlich und örtlich unbeschränktes Nutzungsrecht an den Arbeitsergebnissen der Arbeit einschließlich etwaiger Schutz- und Urheberrechte eingeräumt.

Erlangen, 3. Oktober 2014

Tim Grocki

Zusammenfassung

Zusammenfassung auf Deutsch Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

Abstract

Zusammenfassung auf Englisch Dies hier ist ein Blindtext zum Testen von Textausgaben. Wer diesen Text liest, ist selbst schuld. Der Text gibt lediglich den Grauwert der Schrift an. Ist das wirklich so? Ist es gleichgültig, ob ich schreibe: „Dies ist ein Blindtext“ oder „Huardest gefburn“? Kjift – mitnichten! Ein Blindtext bietet mir wichtige Informationen. An ihm messe ich die Lesbarkeit einer Schrift, ihre Anmutung, wie harmonisch die Figuren zueinander stehen und prüfe, wie breit oder schmal sie läuft. Ein Blindtext sollte möglichst viele verschiedene Buchstaben enthalten und in der Originalsprache gesetzt sein. Er muss keinen Sinn ergeben, sollte aber lesbar sein. Fremdsprachige Texte wie „Lorem ipsum“ dienen nicht dem eigentlichen Zweck, da sie eine falsche Anmutung vermitteln.

Inhaltsverzeichnis

1	Einleitung	1
2	Browserfingerprinting	3
2.1	Funktionsweise	3
2.2	Verwandte Techniken	4
2.3	Nutzung	6
2.3.1	Ersatz für Cookies	6
2.3.2	Nutzeraktionen beschränken	7
2.3.3	Zusätzliches Absichern von Sessions	7
2.3.4	Accounts an Nutzer binden	8
2.3.5	Das Wiedererkennen von böartigen Nutzern	8
2.3.6	Polizei- und Geheimdienstarbeit	8
2.4	Passives und aktives Fingerprinting	9
2.4.1	Passives Fingerprinting	9
2.4.2	Aktives Fingerprinting	10
2.5	Rechtliches	11
2.6	bekannte Gegenmaßnahmen	12
2.6.1	Selbstbeschreibungen einschränken	12
2.6.2	Standardisierung von Systemen	13
2.6.3	Seitenkanäle schließen	13
2.6.4	Skriptsprachen einschränken	13
2.6.5	Traffic normalisieren	14
2.6.6	Fingerprints ändern	14
2.6.7	Entzug der Kommunikation	15
2.6.8	Fälschung von Fingerprints	15
2.7	Zusammenfassung	15
3	Modell	17
4	Gegenmaßnahmen	19
4.1	Stärken und Schwächen	19
4.1.1	Falsch negative Ergebnisse	19
4.1.2	Falsch positive Ergebnisse	20
4.1.3	Informationsnutzung	20
4.1.4	Nutzung zusätzlicher Informationen	20
4.1.5	Seitenkanäle	20
4.1.6	Statistische Abhängigkeiten	20

4.1.7	Datenübermittlung	21
4.1.8	Verknüpfbarkeit von Fingerprints	21
4.1.9	Vorgangscharakter	21
4.1.10	Clientseitiger Code	22
4.1.11	Serverseitiger Code	22
4.1.12	Kommunikation	22
4.1.13	Nutzung der Fingerprints	22
4.1.14	Manipulation des Zufallsprozesses	23
4.1.15	Schwacher Beweis der Funktionsweise	23
4.2	Untersuchte Gegenmaßnahmen und Strategien	23
4.2.1	Randomisieren von Fingerprints	24
4.2.2	Automatische Browserupdates	24
4.2.3	Deaktivieren von Skriptsprachen	24
4.2.4	Fälschung von beliebigen Fingerprints	25
4.2.5	Koordiniertes Fälschen von Fingerprints	25
4.2.6	Blockieren von Kommunikation	25
4.2.7	Filtern von Kommunikation	25
4.2.8	Kombination verschiedener Maßnahmen	26
4.3	Nicht untersuchte Gegenmaßnahmen und Strategien	26
4.3.1	Illegalisierung von Browserfingerprinting	26
4.3.2	Aufklärung der Nutzer	27
4.3.3	Merkmale verheimlichen	27
4.3.4	Fälschen von Kommunikation	27
4.3.5	Vertrauen in Fingerprintstabilität schwächen	27
4.3.6	Fingerprinterspezifische Aktionen	27
4.3.7	Fingerprintänderungen timen	27
4.3.8	Erkennung von genutzten Merkmalen	27
5	Theoretische Betrachtung	29
5.1	Modell	29
5.2	Randomisieren von Fingerprints	29
5.2.1	Notwendiges Maß der Randomisierung	30
5.2.2	Erkennung des Randomisierers	30
5.2.3	Ignorieren von Merkmalen	31
5.2.4	Zusammenfassung	32
5.3	Skripte deaktivieren	32
6	Experiment	33
6.1	Simulation	33
6.1.1	Aufbau der Simulation	33
6.1.2	Bestimmung der Simulationsparameter	35
6.1.3	Ergebniss ohne Gegenmaßnahmen	35
7	Zusammenfassung	37

EINLEITUNG

Medienresonanz Forschung IPv6 Problematik Unklarheit ob bekämpfbar

BROWSERFINGERPRINTING

Das Browserfingerprinting kann genutzt werden, um Nutzer anhand der Merkmale ihres Browsers voneinander zu unterscheiden oder zu identifizieren. In dieser Arbeit liegt der Fokus auf der Nutzerverfolgung und dementsprechend wird sich in dieser Arbeit hauptsächlich auf diese Unterscheidung und Identifizierung konzentriert.

Um einen Überblick über Browserfingerprinting zu geben, wird zunächst in Abschnitt 2.1 seine Funktionsweise zusammengefasst. In Abschnitt 2.2 werden verwandte Techniken dargestellt und in Abschnitt 2.3 wird die Nutzung des Browserfingerprintings vorgestellt. Anschließend wird in Abschnitt 2.4 das aktive und passive Fingerprinting vorgestellt und auf Merkmale eingegangen, die für das Browserfingerprinting genutzt werden. Abschließend wird in Abschnitt 2.5 kurz auf die Rechtslage eingegangen und in Abschnitt 2.6 bekannte Maßnahmen gegen Fingerprinting dargestellt.

2.1 Funktionsweise

Das Browserfingerprinting ist eine Spezialisierung des Prinzip des Fingerprintings von Programmen. Es hat aber nicht das Ziel Programmtypen zu unterscheiden, sondern Browserinstallationen voneinander zu unterscheiden.

Das Fingerprinten von Programmen ist eine schon länger bekannte Technik, die genutzt wird, um gezielte Angriffe auf Computersysteme durchzuführen [21]. Dabei werden Eigenheiten von Programmen genutzt, um auf Informationen über die eingesetzten Programme schließen zu können. Informationen, die auf diese Art und Weise gewonnen werden können, betreffen zum Beispiel den Webservertyp und genaue Programmversionen [27]. Werden möglichst unveränderliche und nicht unterdrückbare Eigenheiten genutzt und kombiniert, ist der Fingerprint wie der namensgebende Fingerabdruck schwer fälschbar und wird auch unfreiwillig abgegeben.

Beispielsweise kann ein Server über den TCP/IP-Stack gefingerprintet werden, um sein Betriebssystem zu bestimmen [40]. Eigenheiten, die dazu genutzt werden können, sind unter anderem die Sortierung von

Optionen, Muster in den Sequenznummern und Maximalgrößen. Diese Eigenheiten werden von dem Analysewerkzeug Nmap abgefragt, indem Pakete gesendet werden und die Antwort analysiert wird.

Das Fingerprinten eines Browsers funktioniert beim Erheben des Fingerprints ähnlich wie das Fingerprinten eines SSH-Servers und kann ebenfalls genutzt werden, um gezielte Angriffe auf Browser durchzuführen [18, 28, 45]. Ein Browser unterscheidet sich aber von Servern wie einem SSH-Server in drei wichtigen Punkten.

- Erstens wird ein Browser auf einem Clientsystem ausgeführt und ist somit zur Verfolgung von Nutzern brauchbar.
- Zweitens ist ein Browser ein komplexeres System als ein SSH-Server, da der Browser wesentlich mehr Eigenheiten, Einstellungs-, Erweiterungs- und Interaktionsmöglichkeiten hat.
- Drittens kann der Server komplexe Seiten mit Skripten ausliefern, die vom Browser interpretiert werden.

Deshalb lassen sich beim Fingerprinten eines Browsers besonders viele Informationen gewinnen. Stehen genug Informationen zur Verfügung, um eine Browserinstallation von allen anderen Browserinstallationen zu unterscheiden, kann diese sogar identifiziert werden. Da diese Informationen nur dazu genutzt werden Browserinstallationen zu unterscheiden, sind sie auch nützlich, wenn ihr Inhalt ansonsten nicht relevant oder nutzlos ist.

Die Möglichkeit, Browserfingerprinting zur Identifizierung von Browserinstallationen zu nutzen, wurde 2009 von Jonathan Mayer vorgestellt [29] und 2010 in einem Paper von Eckersley weiter untersucht [17]. Eckersley hat ein Modell des Browserfingerprintings formuliert, das in Kapitel 3 vorgestellt wird, und eine Studie durchgeführt, um den Informationsgehalt von Browserfingerprints abschätzen zu können. In dieser Studie wurden 470161 Browserfingerprints gesammelt, von denen 83,6 % einzigartig waren, obwohl die genutzte Fingerprintingmethode nicht optimiert war. Die in der Studie aufgebaute Zufallsverteilung hatte eine Entropie von 18,6 Bit, wodurch zu erwarten ist, dass ein zufälliger Fingerprint mit nur einem von 286777 Fingerprints übereinstimmt. In der Diplomarbeit von Tillmann wurde bestätigt, dass die meisten Browserinstallationen mittels Browserfingerprinting identifizierbar sind [42].

In beiden Arbeiten wurde darauf hingewiesen, dass die Stabilität der Browserfingerprints wichtig ist, sollen Browserinstallationen nicht nur unterschieden, sondern auch wiedererkannt werden. Instabil sind Browserfingerprints, wenn sich die Browserinstallation zum Beispiel durch Updates oder Neukonfigurationen ändert. Um trotz dieser Änderungen Browserinstallationen identifizieren zu können, wurden Algorithmen demonstriert, die Änderungen des Fingerprints entdecken und ausgleichen können.

2.2 Verwandte Techniken

Es gibt Techniken, die mit dem Browserfingerprinting verwandt, aber nicht mit ihm identisch sind. Um Missverständnisse und Unklarheiten zu vermeiden, werden diese Techniken dargestellt, ohne dass in der weiteren Arbeit näher auf diese Techniken eingegangen wird.

Die Biometrie ist für das Fingerprinting namensgebend. Bei dieser werden Merkmale eines Menschen erhoben, um sie später erneut zu messen und den Menschen wiederzuerkennen [25]. Dafür genutzte Merkmale sind zum Beispiel Muster der Haut in den Fingerspitzen oder in der Iris von Menschen. Diese Merkmale verändern sich nicht selbstständig und sind schwer zu manipulieren.

Das Fingerprinting von Programmen im Allgemeinen nutzt die Messung von Verhaltenweisen von Programmen, um auf Informationen über diese zu schließen. Diese Informationen reichen im Normalfall nur für die Unterscheidung von Programmen, der Programmversion oder bestimmten Konfigurationen aus. Einsatzzwecke sind das Vorbereiten von gezielten Angriffen [18, 28, 45] und die Erkennung von bösartiger Software [13].

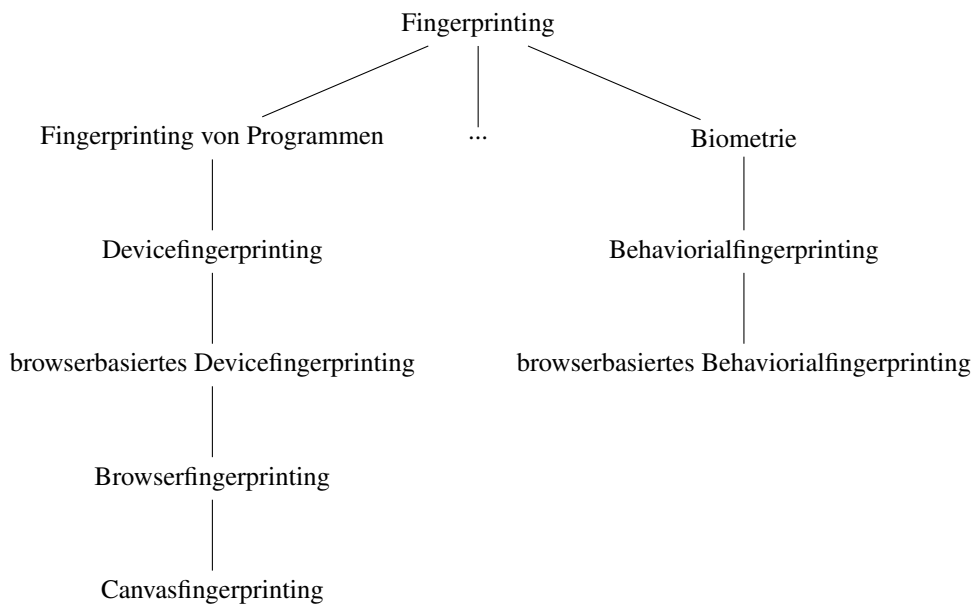


Abbildung 2.1: Hierarchische Darstellung der Fingerprintingvarianten

Das Browserfingerprinting ist ein Spezialfall des Fingerprintings von Programmen, bei dem Browser gefingerprintet werden. Dabei werden sehr viele Informationen preisgegeben, was es erlaubt, nicht nur Browsertypen, sondern auch konkrete Browserinstallationen zu unterscheiden und zu identifizieren. Über die Browserinstallation sollen letztendlich die Nutzer des Browsers unterschieden und identifiziert werden. Diese Form des Fingerprintings wird in dieser Arbeit betrachtet.

Das Devicefingerprinting ist eine direkte Erweiterung des Browserfingerprintings. Dabei wird nicht nur der Browser gefingerprintet, sondern möglichst viele Informationen über den eingesetzten Computer, dessen Betriebssystem und Programme gesammelt. Dazu werden zusätzliche Quellen wie der TCP/IP-Stack in den Fingerprint miteinbezogen [42].

Das Devicefingerprinting ist sehr nahe mit dem Browserfingerprinting verwandt. So enthalten Informationen, die bei dem Browserfingerprinting gesammelt werden, üblicherweise Informationen über das genutzte Betriebssystem oder die Hardware des Nutzers. Dadurch wird Grenze zum Devicefingerprinting bereits überschritten, obwohl nur Informationen aus dem Browser ausgelesen werden. Für diesen Spezialfall gibt es Begriff des „browserbasierten Devicefingerprintings“.

ANMERKUNG: „browserbasiertes Devicefingerprinting“ wäre also der eigentlich korrekte Begriff für das Thema der Arbeit statt „Browserfingerprinting“

Das Behavioralfingerprinting hat nicht die Analyse einer Browserinstallationen zum Ziel, sondern versucht Menschen zu analysieren. Dies geschieht, indem Merkmale des Nutzers selbst und nicht von dessen technischen Systemen erhoben werden. Ein Beispiel dafür ist das Fingerprinten seines Tippmusters oder seiner Mausbewegungen [10, 11]. Um diesen Fingerprint des Nutzers zu messen, können wie im browserbasierten Behavioralfingerprinting im Browser ablaufende Analyseskripte genutzt werden. Das Behavioralfingerprinting ist, obwohl es Browser nutzen kann, der Biometrie und nicht dem Devicefingerprinting zuzurechnen.

Das Canvasfingerprinting ist ein Spezialfall des Browserfingerprintings. Bei diesem werden Merkmale über das Zeichnen und Auslesen von Schriften und Objekten in HTML5-Canvas erhoben. Durch diese

Methode lassen sich viele Informationen gewinnen und das Canvasfingerprinting ist eine oft eingesetzte Fingerprintingtechnik.

Das Fingerprinting aus andern Kontexten wie der Geologie [38], Biologie [35], Medizin [16], Forensik [20] oder auch der Klimaforschung [47] funktionieren größtenteils ebenfalls nach dem Prinzip des Wiedererkennens von Merkmalen. Die dabei genutzten Methoden und verfolgten Ziele sind dabei allerdings sehr unterschiedlich und werden auch in dieser Übersicht nicht weiter behandelt.

2.3 Nutzung

Die Nutzung von Browserfingerprinting wurde in der von Nick Forakis und anderen durchgeführten Studie „Cookieless Monster: Exploring the Ecosystem of Web-based Device Fingerprinting“ untersucht [36]. Bei dieser Studie wurde ein Crawler genutzt, um Webseiten zu finden, die Fingerprinting einsetzen. Diese Studie wurde im Jahr 2013 durchgeführt und auf 40 der 10000 nach dem Alexa-Rating am meisten besuchten Webseiten Fingerprintingskripte gefunden. Mit derselben Technik wurden ebenfalls 2013 in 404 der Top 1000000 auf Javascript basierendes Fingerprinting und in 97 der Top 10000 flashbasiertes Fingerprinting gefunden [9]. Diese Zahl ist aber eine untere Grenze, da mit dem eingesetzten Crawler nur bestimmte Browserfingerprintingskripte erkannt werden konnten und nicht alle Unterseiten überprüft wurden. In einer neueren Studie [8] wurde festgestellt, dass über 5 % der Top 100000 Webseiten Canvasfingerprinting, ein Spezialfall des Browserfingerprintings, einsetzen.

Die Frage, wofür die gefundenen Browserfingerprintingskripte genutzt werden, konnte allerdings nicht abschließend geklärt werden. Untersucht wurde aber, auf welcher Art von Webseite Browserfingerprintingskripte gefunden wurden. Die stärksten Kategorien waren Spam, Pornografie und bösartige Seiten, wobei auch dies als Schätzung zu betrachten ist.

Bekannte Nutzungsmöglichkeiten dieser Form der Nutzerwiederidentifizierung sind

- die Nutzung als Ersatz für Cookies,
- das zusätzliche Absichern von Sessions,
- die Bindung von Nutzern an Services,
- das Identifizieren von bösartigen Nutzern und
- die Polizei- oder Geheimdienstarbeit.

Diese Nutzungsmöglichkeiten und ihre prinzipiellen Funktionsweisen werden in den nächsten Abschnitten vorgestellt.

2.3.1 Ersatz für Cookies

Es gibt zwar verschiedene Arten, Daten in einem Browser abzulegen, die Cookies direkt ersetzen können [31, 39] und sogar wie bei „evercookie“ gesammelt ansprechbar sind [26], aber Browserfingerprinting wird trotzdem genutzt, um Cookies zu ersetzen [36]. Im Gegensatz zu Cookies wird dabei das Ablegen von Daten in dem Browser vermieden. Deshalb hat der Nutzer bei Fingerprints keine Möglichkeit diese wie Cookies zu löschen und die Wiederidentifizierung über Browserfingerprints hinterlässt nicht zwangsweise Spuren im Browser. Wenn Privatmodi von Browsern lediglich verhindern, dass Daten im Browser abgelegt werden, sind diese

Diese Möglichkeit, Cookies zu ersetzen, kann wie Cookies selbst für verschiedene Zwecke eingesetzt werden. Eine dieser Möglichkeiten ist die Erstellung von Nutzerprofilen zu Verkaufs- oder Werbezwecken.

Browserfingerprinting kann auch in Kombination mit Cookies genutzt werden, um Nutzer trotz des Löschens aller Cookies wiederzuerkennen und die Cookies wiederherzustellen. Eckersley gibt an [17], dass für

eine solche Cookie Regeneration 15-20 Bit Informationen ausreichen, wenn weitere Informationen wie IP-Adressen hinzugezogen werden. Es wird eine Zunahme der Nutzung dieser Technik erwartet, da Tracking-Unternehmen Cookies nutzen, und Cookies zunehmend als Problem behandelt und vermieden werden. Ein ähnliches Regenerationsystem wird für die Regeneration von Cookies über Flashcookies genutzt [8].

Eine zur Cookie Regeneration ähnliche Technik wurde 2012 untersucht [48]. Bei dieser wurden lediglich der Useragent und die IPs der Nutzer kombiniert, um diese zu identifizieren. Kombiniert wurden diese beiden Eigenschaften auf 20,29 Bit Entropie geschätzt, was höher ist als die ursprüngliche Schätzung von Eckerlsey für das Browserfingerprinting insgesamt [17]. Diese Studie wurde den Nutzern von Bing und Hotmail über den Zeitraum von mehreren Monaten durchgeführt und hat somit einen großen Umfang von mehreren hundert Millionen Nutzern.

2.3.2 Nutzeraktionen beschränken

Auf Datingseiten oder Umfrageseiten existiert ein Interesse, Mehrfachregistrierungen komplett zu verhindern, da im Fall von mehreren Accounts für einen Nutzer Missbrauch vermutet werden kann [9, 36]. Auch Zeitungen könnten diese Möglichkeit nutzen, um Nutzer eine gewisse Anzahl von Artikeln lesen zu lassen und erst danach eine Gebühr zu verlangen [9].

Sollen bestimmte Aktionen für jeden Nutzer auf eine oder mehrere Nutzungen eingeschränkt werden, kann versucht werden dies über den Fingerprint seines Browserden Fingerprint seines Browsers durchzusetzen. Dazu können einfach die Fingerprints der Nutzer gespeichert werden und der Fingerprint nach Beenden der Aktion als verbraucht markiert oder ein dem Fingerprint zugeordneter Zähler inkrementiert werden. Hat ein Nutzer einen bereits verbrauchten Fingerprint, kann verweigert werden, die angeforderte Aktion durchzuführen.

2.3.3 Zusätzliches Absichern von Sessions

Sollen eine Reihe von Aktionen ausgeführt werden, die durch ein Login geschützt sind, wird oft eine Session genutzt, um das ursprüngliche Passwort nur einmal zu übertragen. Um den Nutzer wiederzuerkennen, wird zu Beginn der Session ein Geheimnis generiert, dass sich Client und Server teilen. Durch Vorlage dieses Geheimnisses kann sich der Client gegenüber dem Server ausweisen und auf die mit der Session verknüpften Daten zugreifen.

Beim Sessionhijacking erfährt oder errät ein Angreifer dieses Geheimnis und nutzt es, um sich gegenüber dem Server auszuweisen. Um Nutzer gegen einen solchen Angriff zu schützen, kann der Server zusätzlich den Fingerprint des Clients speichern und überprüfen [43]. Ändert sich der Fingerprint auffällig stark, kann angenommen werden, dass eine unzulässige Übertragung der Session stattgefunden hat und die Session kann geschlossen werden.

Die Sicherheit eines solchen Systems beruht auf den Annahmen, dass der Fingerprint dem Angreifer unbekannt ist oder der Angreifer den Fingerprint nicht reproduzieren kann und dass der Angreifer nicht zufällig den Fingerprint des Opfers hat. Da nicht erwiesen ist, dass diese Annahmen zutreffen, kann Sessionhijacking nicht ausgeschlossen werden, aber immerhin die Komplexität eines solchen Angriffs erhöht werden. Diese Sicherheitsmaßnahme profitiert zwar von einer großen Menge an preisgegebenen Information, funktioniert aber auch mit geringen Mengen preisgegebener Information.

Gelingt es einem Angreifer, den legitimen Fingerprint zu fälschen, versagt dieser Schutz und der Angreifer kann mit seinem Angriff fortfahren. Ändert sich der Fingerprint eines legitimen Nutzers zu stark und wird die Session unberechtigt geschlossen. Allerdings ist der Nutzer lediglich gezwungen, sich erneut einzuloggen und es ist also nur geringer Schaden zu erwarten.

2.3.4 Accounts an Nutzer binden

Soll die Nutzung eines Accounts auf einen oder wenige Nutzer beschränkt werden, kann versucht werden dies über deren Fingerprint ihres Browsers durchzusetzen [36]. Dazu speichert der Server einen oder mehrere Fingerprints, die mit den Accounts verknüpft sind, und kann nun bei Login-Versuchen überprüfen, ob der Fingerprint des Clients mit den verknüpften Fingerprints übereinstimmt. Um Fingerprints mit einem Account zu verknüpfen, kann zum Beispiel der Fingerprint des ersten Logins gespeichert oder zusätzliche Authentifikationsmethoden genutzt werden. Ist der Fingerprint nicht mit dem Account verknüpft, kann der Login verweigert, zusätzliche Authentifikationsmethoden gefordert oder ein Einbrucherkennungssystem benachrichtigt werden.

Dies kann im Hochsicherheitsbereich genutzt werden, bei dem angenommen werden kann, dass der Nutzer seinen Account nur von einem Gerät nutzt. Existieren zusätzliche Authentifikationsmethoden, kann auch auf Änderungen des Fingerprints von legitimen Nutzern eingegangen werden. Ein Beispiel für einen solchen Hochsicherheitsbereich sind Onlinebezahlssysteme [36] oder Banken. Aber auch in Bereichen mit niedrigen Sicherheitsanforderungen, wie dem bezahlten Mediasstreaming, kann eine Bindung von Accounts an Browser angewendet werden. Bei diesem haben die Nutzer ein Interesse an der Mehrfachnutzung eines Accounts, die Anbieter hingegen ein Interesse daran, dass möglichst viele Accounts erstellt werden. Um zu verhindern, dass sich eine große Menge von Nutzern unter einem Account einloggt und sich ein Nutzer trotzdem von mehreren Geräten einloggen kann, wird einer kleinen Menge von Fingerprints an den Account gebunden.

Genau wie bei dem zusätzlichen Absichern von Sessions ist die vollständige Sicherheit dieses Mechanismus nicht erwiesen, da hier dieselben Annahmen zugrunde liegen. Eine fehlerhafte Erkennung eines legitimen Nutzers als Angreifer kann hier allerdings schwerwiegendere Konsequenzen für diesen Nutzer haben, wenn diesem zum Beispiel der Login dauerhaft verweigert wird.

2.3.5 Das Wiedererkennen von bössartigen Nutzern

Bössartige Nutzer wie Hacker, Scammer, Spammer oder Trolle hinterlassen normalerweise Spuren wie IP-Adressen auf dem Server. Benutzt der Angreifer einen Browser, kann der Browserfingerprint eine der hinterlassenen Spuren sein.

Diese Browserfingerprints können gesammelt, in Datenbanken zusammengefasst und eventuell genutzt werden, um bössartige Nutzer anhand dieser Datenbank zu erkennen [9]. Diesen Nutzern kann zum Beispiel die Nutzung der Seite verweigert werden oder es kann ein Einbrucherkennungssystem über deren Besuch benachrichtigt werden. Diese Datenbanken könnten auch zwischen mehreren Parteien geteilt und ausgetauscht werden, wenn der zum Erstellen des Fingerprints genutzte Fingerprintalgorithmus allen Seiten bekannt ist.

Dieses System versagt, wenn der Angreifer seinen Fingerprint verändert und somit nicht mehr erkennbar ist. Teilt sich ein Angreifer einen Fingerprint mit einem Unbeteiligten, wird dieser fälschlicherweise ebenfalls in die Datenbank von Angreifern aufgenommen und entsprechend behandelt. Sollen diese Fehler möglichst verhindert werden, benötigt dieses System eine möglichst große Menge an Entropie. Dies ist besonders dann der Fall, wenn die Datenbank geteilt und die Menge der betroffenen Nutzer so vergrößert wird.

2.3.6 Polizei- und Geheimdienstarbeit

Browserinstallationen identifizieren oder auch nur unterscheiden zu können, wäre für Polizei- und Geheimdienstarbeit nützlich. Dadurch könnten zum Beispiel Webseitenbesuche mit im Nachhinein beschlagnahmten Betriebssysteminstallationen in Verbindung gebracht werden. Mehrfachtäter könnten über Browserfingerprints erkannt und Taten in Zusammenhang gestellt werden. Die wohl mächtigste Nutzungsmöglichkeit ist das Deanonymisieren von Nutzern von Anonymisierungsnetzwerken wie TOR. Dies kann zum Beispiel geschehen, indem der Browserfingerprint nach Verlassen der Anonymisierungsnetzwerkes und in einer Situation gemessen wird, bei der der Zugriff dem Nutzer zugeordnet werden kann.

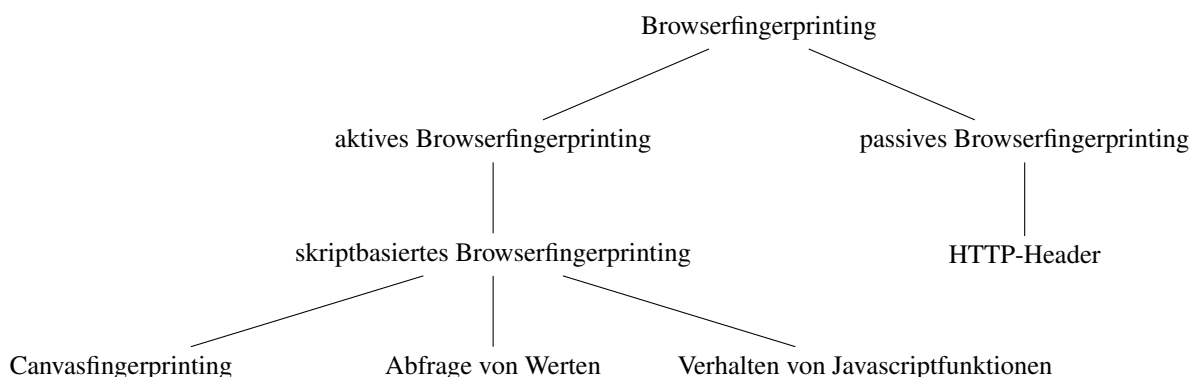


Abbildung 2.2: Hierarchische Darstellung der Formen des Browserfingerprintings

Da angenommen wird, dass Polizei- oder Geheimdienste Browserfingerprinting nutzen oder zumindestens die Möglichkeit dazu haben, bietet das Anonymisierungswerkzeug TOR über den TOR-Browser ein gegen Browserfingerprinting abgehärteten Browser. Die für die Polizeiarbeit relevante Frage der Beweiskraft von Browserfingerprints ist dabei noch ungeklärt.

2.4 Passives und aktives Fingerprinting

Browserfingerprinting kann anhand der genutzten Methoden zum Erheben von Fingerprints in aktives und passives Browserfingerprinting unterscheiden werden. Für das Fingerprinting von Programmen wird eine Definition für aktives und passives Browserfingerprinting genutzt [41]. In dieser Definition ist Fingerprinting passiv, wenn die Kommunikation zwischen Client und Server nicht verändert wird, und ansonsten aktiv.

Als Unterscheidung zwischen aktivem und passivem Browserfingerprinting findet sich auch eine Definition, dass aktives Fingerprinting clientseitige Skriptsprachen wie Javascript nutzt, passives hingegen nicht [42]. Das Provozieren von unterschiedlichen Verhaltensweisen im Browser, ohne clientseitige Skripte zu nutzen, ist nach dieser Definition dem passiven Browserfingerprinting zuzuordnen, obwohl es eigentlich ein aktives Vorgehen ist. Dies wird zum Beispiel bereits genutzt, um CSS-Layouts auf verschiedene Browser anzupassen [1]. Dazu werden bereits Unterschiede in Syntaxparsing der Browser genutzt, um ohne Javascript zwischen Browsern differenzieren zu können. Um diesen Fall unter aktives Fingerprint fallen zu lassen, wird diese Definition nicht genutzt.

2.4.1 Passives Fingerprinting

Beim passiven Fingerprinting werden Informationen in den Fingerprint einbezogen, die während der normalen Nutzung eines Dienstes wie einer Webseite auftreten. Typische Werte, die mittels passiven Fingerprintings ausgelesen werden, sind in Tabelle 2.1 aufgelistet.

Accept	Vom Browser unterstützte und erwünschte Datenformate
Accept-Language	Sprache, in der die Webseite generiert werden soll
Accept-Encoding	Vom Browser unterstützte Codecs
Accept-Charset	Unterstützte und erwünschte Zeichenkodierungen
Connection	Gewünschte Art der weiteren Verbindung
Useragent	Kurze Selbstbeschreibung oder Name des Browsers

Tabelle 2.1: Passive Fingerprintingmerkmale [15]

Diese Merkmale können auch beim aktiven Fingerprinting ausgelesen werden, deswegen fallen beim passiven Fingerprinting im Vergleich zum aktiven Fingerprinting weniger Informationen an. Die Menge der Entropie der beim passiven Fingerprinting kann nur geschätzt werden, aber als Untergrenze für die Entropie des Useragents liegen Schätzungen für 6,31 Bit von Broenink, 8,10 Bit von Boda, 10 Bit von Eckersley und 11,59 Bit von Yen und anderen vor [44].

Eine Eigenschaft des passiven Browserfingerprintings in der verwendeten Definition ist, dass es keine Spuren bei dem betroffenen Browser oder im Traffic hinterlässt. Deshalb kann passives Browserfingerprinting vom Client nicht aufgrund solcher Spuren erkannt werden. Aus diesem Grund konnte in der Studie „Cookieless Monster: Exploring the Ecosystem of Web-based Device Fingerprinting“ auch nur das aktive Browserfingerprinting erkannt und untersucht werden.

Eine weitere Eigenschaft, die passives Fingerprinting haben muss, ist, dass nicht nur der Server oder Client, sondern auch dritte Parteien Fingerprints können. Dies ist möglich, wenn Drittparteien Kenntnis über die unverschlüsselte Kommunikation zwischen Client und Server erlangen, da diese für passives Browserfingerprinting den Traffic nicht verändern müssen. Zum Browserfingerprinting durch Drittparteien konnten leider keine Untersuchungen gefunden werden, im Hinblick auf einen Geheimdienst als Drittpartei ist diese Möglichkeit aber relevant.

2.4.2 Aktives Fingerprinting

Beim aktiven Fingerprinting wird die Kommunikation zwischen Server und Client so gestaltet, dass der Client mehr Informationen über sich preisgibt, als er es normalerweise tun würde. Dieses Gestalten der Kommunikation kann zum Beispiel geschehen, indem das zu fingerprintende System einfach nach weiteren Informationen, wie unterstützten Aktionen gefragt wird [23]. Weitere Eigenheiten können ausgeforscht werden, indem Maximalgrößen getestet werden [14] oder Syntax verwendet wird, die undefiniertes oder nicht einheitliches Verhalten aufweist [6].

Moderne Browser bieten normalerweise die Möglichkeit, Skripte im Browser ausführen zu lassen. Diese Sprachen bieten sehr weitreichende Möglichkeiten, Informationen über die Browserinstallation zu gewinnen. Einfache Methoden fragen diese Informationen einfach bei dem Browser an. Komplexere Methoden sind zum Beispiel das pixelgenaue Analysieren von Schriftdarstellungen [32] und das Benchmarken von Javascriptfunktionen, um auf die genutzte Javascriptengine zu schließen [33, 34]. Für das Zurücksenden der gesammelten Daten kann auf dafür vorgesehen Funktionen wie AJAX [17] zurückgegriffen werden, aber auch andere verstecktere Kanäle könnten genutzt werden [36].

Einer der bekanntesten von Browsern unterstützten Skriptsprachen ist Javascript. Alleine diese war im Jahr 2010 von 99 % der menschlichen Nutzer aktiviert [9] und es gibt weitere Sprachen wie Flash, Silverlight oder Java, die zum Fingerprinten geeignet sind. Beim Browserfingerprinting kann also davon ausgegangen werden, dass es möglich ist, solche Skripte einzusetzen, ohne allzu viele Nutzer auszulassen. Um sicherzustellen, dass auch tatsächlich alle Nutzer Skripte aktiviert haben, könnte der Webseiteninhalt über diese geladen und so alle Nutzer mit deaktivierten Skripten ausgeschlossen werden.

Browsermerkmale, von denen bekannt ist, dass sie für aktives Browserprinting verwendet werden können, sind unter in Tabelle 2.2 aufgelistet.

Anmerkung: gerne hätte ich hier noch aktive nicht skript attribute reingenommen, aber ich finde da wenig konkretes

Anmerkung: Seitenweise Details gibts unter cs.gmu.edu/~yhwang1/INFS612/2013_Spring/Projects/Final/2013_Spring_PGN_5_f ich hab das aber nicht detailliert reingenommen.

Das Browserfingerprinting mittels Skripten steht, wohl aufgrund der Menge an erheblichen Informationen und der breiten Verfügbarkeit von Skripten, im Fokus von Industrie und Forschung. Das Interesse der Industrie ist daran erkennbar, dass viele Browserfingerprintingbibliotheken sehr stark Javascript, Flash oder Silverlight nutzen [36]. Wie stark das Interesse der Forschung an skriptbasierten Browserfingerprinting ist, ist daran erkennbar, dass der Fingerprintingmechanismus, der in der Diplomarbeit von Tillmann zur Untersuchung von Browserfingerprints eingesetzt wurde, ohne aktivierte Skripte noch nicht einmal lauffähig war

Appname [42]	Eine kurze Selbstbeschreibung oder Name des Browsers
Javascriptversion [15]	Die Javascriptversion
Hardwareinformationen [15]	z.B. Prozessorarchitektur oder Bildschirmauflösung
Charset [15]	Vom Browser unterstützte Zeichencodierung
Sprache [15]	Die Spracheinstellung des Browsers
Cookieunterstützung [15]	Die Unterstützung von Cookies und Supercookies
Unterstützung von Skripten [15]	Die Unterstützung von Sprachen wie Java, Flash oder Silverlight
Zeitzone [15]	Die Zeitzoneneinstellung des Browsers
Farben [42]	Die eingestellten Farben für Buttons, aktivierte Links oder ähnliches
Plugins [17, 42]	Die installierten Plugins
Mimetypes [42]	Dem Browser bekannte Mimetypes
Schriften [17, 42]	Die dem Browser zur Verfügung stehenden Schriften
Schriftdarstellungen [3, 32]	Analyse der Darstellungsweisen von Schriften durch den Browser
Benchmarks [33, 34]	Die Dauer von bestimmten Aktionen
Privatsphäreinstellungen [36]	Einstellungen wie der Do-Not-Track-Header
Konstanten [36]	Mathematische Konstanten des Browsers
Spezialfähigkeiten [36]	Fähigkeiten und Funktionen, die nur manche Browser haben

Tabelle 2.2: aktive Fingerprintingmerkmale

und somit nicht skriptbasiertes Browserfingerprinting vernachlässigt wurde [42]. Ebenso konnte der Crawler, mit dem in der Studie „Cookieless Monster: Exploring the Ecosystem of Web-based Device Fingerprinting“ nach Browserfingerprinting gesucht wurde, nur auf Skripten basierendes Browserfingerprinting erkennen [36].

2.5 Rechtliches

Das Browserfingerprinting zur Nutzerverfolgung wird über verschiedene angemeldete Firmen und Werbenetzwerken kommerzialisiert. Daran ist erkennbar, das Browserfingerprinting nicht in einer komplett illegalen Umgebung stattfindet und somit auch die detaillierte Rechtslage interessant ist. Bei den verwandten Cookies haben diese Regelungen, ähnlich wie der Do-Not-Track-Header allerdings kaum Effekt [8].

Ein zentraler Faktor in Tillmanns rechtlicher Bewertung [42] ist die Frage, ob beim Browserfingerprinting personenbeziehbare oder pseudonymisierte Daten vorliegen. Liegen personenbeziehbare oder personenbezogene Daten vor, fallen diese in Deutschland unter das Bundesdatenschutzgesetz. Dieser Schutz kann die explizite Einwilligung des Nutzers vor der Datensammlung oder das Verbot, Datenbestände zusammenzuführen, beinhalten. Die Wichtigkeit dieses Punktes wird dadurch bestärkt, dass auch Firmen unabhängig von deutschem Recht, beteuern keine personenbezogene Informationen zu speichern, obwohl sie Browserfingerprinting einsetzen [9]. Es wird auch befürchtet, dass diese Argumentation ausgeweitet werden könnte, um Datenschutzgesetze zu umgehen. Dazu könnten Daten mit einem Browserfingerprint und nicht mit einem Nutzer verknüpft gespeichert werden.

In einer Untersuchung zu Browserfingerprinting wurde festgestellt, dass die Nutzer üblicherweise nicht über die beim Browserfingerprinting vorgenommene Datensammlung informiert werden [9]. Beim Browserfingerprinting, dass durch Drittparteien ausgeführt wird, könnte dabei auch die Frage Probleme bereiten, wer informieren sollte. Der Nutzer kann grundsätzlich auf der einbindenden oder eingebundenen Seite informiert werden. Da das Einbinden von Drittparteien für den normalen Nutzer oft nicht erkennbar ist, sollte intuitiv die einbindende Seite irgendeinen Hinweis auf das Browserfingerprinting beinhalten. Die einbindende Seite hat aber nicht notwendigerweise das Wissen darüber, das Browserfingerprinting eingesetzt wird.

Für Cookies gibt es in der EU Vorschriften, die die Zustimmung des Nutzers zum Speichern von Daten in Browsern erfordern [30]. Dieses wird allerdings vom Browserfingerprinting vermieden. Wird Browserfingerprinting verwendet, um Cookies zu regenerieren, muss davon ausgegangen werden, dass der Browser

des Nutzers keine Cookies erlaubt oder Cookies gelöscht werden. Der Nutzer drückt somit seinen Willen aus, nicht wiedererkannt zu werden und eine Nutzung zu diesem Zweck kann illegal sein. Das Wiederherstellen von Cookies hat auch bereits zu Gerichtsverfahren mit einer Einigung in der Höhe von 500000 \$ geführt [8].

Für die rechtliche Betrachtung ist auch interessant, dass das Regenerieren von Cookies mittels Browserfingerprinting seit 2011 in den USA zum Patent angemeldet ist [24]. Wäre dieses Patent gültig und würde verteidigt werden, könnte dies den Einsatz von Browserfingerprinting erheblich verkomplizieren, auch wenn der Nutzer informiert und einverstanden wäre. Welchen Effekt ein solches Patent hätte, kann an dieser Stelle aber nicht eingeschätzt werden.

2.6 bekannte Gegenmaßnahmen

Zur Abwehr von Browserfingerprinting und dem Fingerprinten von Programmen existieren mehrere Vorschläge zur Vorgehensweise. Diese Gegenmaßnahmen wurden, um die Übersicht und Referenzierbarkeit zu erhöhen, in Kategorien unterteilt. Es besteht dabei allerdings keinen Anspruch auf Vollständigkeit. Als Quellen für diese Gegenmaßnahmen wurden Veröffentlichungen, Diskussionen, Privatsphären-Plugins und das Tor-Browser-Bundle herangezogen.

2.6.1 Selbstbeschreibungen einschränken

Eine intuitive Vorgehensweise, um die vom Browser preisgegebene Selbstinformation zu reduzieren, ist die Menge der vom Browser freiwillig herausgegebenen Daten klein zu halten. Die Geheimhaltung von Informationen wie Sprachpräferenzen oder installierte Schriften, verhindert die Nutzung von Features, die auf diesen Informationen basieren.

Das führt allerdings durch das „Privacy paradox“ nicht zwingend zum Erfolg, sondern kann auch den eigentlichen Zielen entgegenwirken [17]. Dies lässt sich darauf zurückführen, dass das Geheimhalten einer Information selbst wieder eine Information ist, die zur Unterscheidung von Browserinstallationen verwendet werden kann. Impliziert das Geheimhalten von Information mehr Information als die geheim gehaltene Information, gibt der Browser also insgesamt mehr Information über sich preis. Dies kann sogar zur Identifizierbarkeit einer Browserinstallation ausreichen, wenn eine Information nur von dieser geheim gehalten wird. Der Effekt des „Privacy paradox“ kann allerdings verringert werden, indem sichergestellt wird, dass viele Browser diese Information geheim halten. Auf diese Weise verringert sich die Menge der durch Geheimhaltung preisgegebene Information und muss auf 0 Bit fallen, wenn alle Browser diese Information geheim halten.

Es existieren verschiedene Ansätze, um die vom Browser freiwillig herausgegebene Daten zu verringern. Drei dieser Wege werden hier als Beispiel dargestellt.

Eckersley hat vorgeschlagen, dass Browser und Add-ons statt Mikroversionsnummern nur grobe Versionsangaben preisgeben könnten [17]. Vermutlich aus Debugginggründen werden Versionsnummern teilweise mit Revisions oder spezifischen Builds angegeben. Da Versionsangaben von Browser oder Plugin-Herstellern festgelegt werden, würden grobe Versionsangaben auch automatisch von allen Nutzern übernommen. Eckersley weist dabei auch darauf hin, dass ein dem widersprechendes Interesse gibt, genaue Versionsnummern für Debugging nutzen.

Eckersley hat zudem vorgeschlagen Informationen wie die installierten Schriftarten nicht als Liste, sondern nur noch auf Anfrage für spezifische Werte herauszugeben [17]. Das hätte zur Folge, dass obskure Schriftarten nicht oder nur durch Prüfung einer großen Liste von Schriftarten in das Fingerprinting einbezogen werden können. Dadurch, dass diese Schriftarten obskur sind, muss ihre Installation eine große Menge von Informationen tragen. Auch die Information, die in der Sortierung solcher Listen liegt, würde auf diese Weise nicht mehr freigegeben werden.

2.6.2 Standardisierung von Systemen

Identische Browserinstallationen können durch eine Messung also einem Browserfingerprint nicht unterschieden werden. Je ähnlicher sich Browserinstallationen sind, desto genauer muss die Messung sein, um die Unterschiede zu erkennen. Findet eine breite Standardisierung von Browserinstallationen statt, indem zum Beispiel der Installationsvorgang standardisiert wird, geben diese nur wenig Informationen über sich preis. Je strikter ein System standardisiert ist, desto stärker ist der Nutzer beeinträchtigt, da dieser keine freie Wahl zwischen Programmen oder Konfigurationsoptionen treffen kann.

Mit diesem Effekt erklärt Eckersley die im Vergleich zu anderen Browsern geringe Informationsmengen, die von manchen Browsern preisgegeben werden [17]. Konkret nennt Eckersley Browser von Smartphones wie dem iPhone, die bereits vorinstalliert sind und kaum konfigurierbar sind. Ein anderes Beispiel sind Maschinenklone, bei denen eine Referenzinstallation eines Betriebssystems auf viele Computer kopiert wird [17].

Das TOR-Browser-Bundle nutzt diesen Effekt, um möglichst wenig Informationen über Browserinstallationen preiszugeben. Dieses versucht nicht, vorhandene Browser anzupassen und wird nicht über die üblichen Installationswege installiert. Stattdessen ist das TOR-Browser-Bundle ein stark in sich abgeschlossenes Paket und benötigt auf Linux keine weitere Installation oder Konfiguration. Durch die Empfehlung, aus verschiedenen Gründen keine anderen Browser im Tor Netzwerk zu nutzen, wird versucht ein Standard zu setzen, der von einer großen Gruppe von Personen genutzt wird [15]. Dieses Prinzip kann auch weitergeführt werden, indem auf standardisierte Betriebssysteminstallationen zurückgegriffen wird, um mit diesen in einem Anonymity-Set zu liegen. Dies kann zum Beispiel geschehen, indem Live CDs genutzt werden, die keinerlei Installation oder Konfiguration benötigen und großen Mengen verbreitet sind.

2.6.3 Seitenkanäle schließen

Informationen über einen Browser können von diesem nicht nur absichtlich, sondern auch unbeabsichtigt über Seitenkanäle übermittelt werden [46]. Dies ist gewöhnlicherweise nicht nur unbeabsichtigt, sondern auch unerwünscht, da auf diese Weise geheime Daten übermittelt werden könnten. Würden diese Seitenkanäle eliminiert werden, wären für den Nutzer keine Einbußen zu befürchten, da Seitenkanäle per Definition, nicht zum gewünschten Funktionsumfang eines Programmes gehören.

Können Skriptsprachen verwendet werden, um Seitenkanäle auszulesen, existieren weitreichende Möglichkeiten, an Informationen zu gelangen. So können zum Beispiel pixelgenaue Analysen von Schriften durchgeführt werden [32], Systemstandards für Farben ausgelesen werden und Javascriptbenchmarks durchgeführt werden. Die Komplexität von Browsern und die Größe der den Skripten zu Verfügung gestellten APIs macht eine Eliminierung eines relevanten Teiles dieser Seitenkanäle sehr aufwendig.

2.6.4 Skriptsprachen einschränken

Skriptsprachen wie Javascript erlauben, wie in Sektion 2.4.2 dargestellt, viele Informationen aus Seitenkanälen und API-Abfragen zu gewinnen. Diese Skriptsprachen erlauben nicht nur das Erheben von Daten, sondern bieten auch die Möglichkeit, diese Informationen an einen Server zu senden.

Ein Ansatz, dies zu verhindern, ist das Deaktivieren von Skripten [36]. Dadurch gehen alle auf Skripten basierenden Nutzungsmöglichkeiten verloren und Webseiten können sogar unbenutzbar werden. Die Analysemöglichkeiten werden dadurch allerdings so stark eingeschränkt, dass dies eine effektive Gegenmaßnahmen darstellt und komplette Fingerprintingbibliotheken nicht lauffähig sind.

Ein anderer Ansatz, der auch im TOR-Browser verwendet wird, ist die Reduierung der Mächtigkeit der Skriptsprachen [5, 7]. Damit würde letztlich ein neuer Sprachdialekt erschaffen, der in Hinblick auf geringe Informationsfreigabe gewählt werden könnte. Die Analyse des Browsers könnte dadurch behindert werden, dass nur wenig Information absichtlich preisgegeben werden und es wenig Gelegenheiten gibt, Seitenkanäle auszulesen. Könnte verhindert werden, dass eine Skriptsprache mit einem Server kommunizieren kann, ist die Art und Menge der gesammelten Information sogar irrelevant, da diese nicht an den Server übermittelt

werden könnte. Dafür müssten aber nicht nur die dafür vorgesehen Funktionen, sondern alle zur Kommunikation geeigneten Seitenkanäle entfernt werden. Diese Lösung würde dem Nutzer die Möglichkeit geben, Webseiten mit einfachen Skripten zu nutzen und trotzdem die Menge an Informationen zu reduzieren, die über die Browserinstallation in Erfahrung gebracht werden könnten.

In dem TOR-Browser-Bundle wird eine Zwischenlösung zwischen der kompletten Geheimhaltung und der kompletten Preisgabe von Informationen angewendet. Dabei dürfen nur eine feste Menge von Sprachen über die dafür vorgesehene Javascript-API abgefragt werden [9]. Bei dieser Art der Geheimhaltung ist allerdings zu beachten, dass die Abfrage von Informationen über APIs nur ein Weg ist, an Informationen zu gelangen. Wird die Information auf anderem Weg in Erfahrung gebracht, gibt der Browser nicht nur die Information preis, die geheim gehalten werden sollte, sondern auch die Information, dass dies versucht wurde.

Ein Beispiel für den gegenteiligen Effekt, also die Preisgabe von Informationen durch Erweiterungen, ist das Einführen von Canvas in HTML-5. Bei diesem können Schriften und Formen gezeichnet und die Darstellung pixelgenau analysiert werden. Diese Erweiterung von Skriptsprachen wird erfolgreich zum Fingerprinten verwendet. [8]. Im TOR-Browser wird dies allerdings verhindert, indem ein leeres Bild zurückgegeben wird.

2.6.5 Traffic normalisieren

Eine Möglichkeit, die vorgeschlagen wurde, um TCP/IP-Fingerprinting von Servern zu verhindern, ist das Normalisieren des Traffics [40]. Bei dieser Methode wird der von den Servern generierte Traffic abgefangen, in eine abstrahierte Form übertragen und der Traffic auf Basis der abstrahierten Form wiederhergestellt. Bei dieser für TCP/IP semantisch unbedeutenden Transformation sollen Informationen, die auf die in den Servern eingesetzte Software schließen lassen, verloren gehen.

Wollte man dieses Konzept auf Browserfingerprinting übertragen, müsste der HTTP-Traffic abgefangen werden und eine Abstrahierung des HTTP-Traffics möglich sein. Da die Verschlüsselung von HTTP-Traffic serverseitig erzwungen werden kann, ist dieses Konzept nicht direkt auf Browserfingerprinting übertragbar.

2.6.6 Fingerprints ändern

Soll ein Browser nicht wiederidentifiziert werden, kann versucht werden die Browserinstallation so stark zu verändern, dass sie nicht wiedererkannt werden kann. Wird die Browserinstallation zwischen jeder versuchten Wiederidentifizierung so geändert, dass diese nicht mehr erkannt werden kann, ist die Menge der preisgegebenen Informationen und die Einzigartigkeit der Browserinstallation unwichtig. Kann die Veränderung der Browserinstallation vom Server erkannt werden, könnte dies genutzt werden, um die veränderte Information zu ignorieren und die Browserinstallation wiederzuerkennen.

Mit dem Browser-Plugin „Firegloves“ gab es einen Ansatz, dieses Prinzip zu nutzen, um ein Plugin gegen Browserfingerprinting zu erstellen. Dabei wurden Browserattribute randomisiert und so der Fingerprint geändert. Zu diesem Projekt sind aber leider nur noch Referenzen und Beschreibungen zu finden [12, 37], weswegen auf eine detaillierte Beschreibung verzichtet werden muss.

Mit „PriVaricator“ [37] existiert ein weiterer Ansatz, der auf der Randomisierung von Browsermerkmalen beruht. Bei diesem Plugin wird die über Javascript ermittelte Liste der installierten Schriftarten und Plugins randomisiert, um einen neuen Fingerprint zu erhalten. Ein Orakel für ein in der Industrie eingesetztes Fingerprintingskript, konnte auf diese Weise getäuscht werden und hat die randomisierten Fingerprints als verschiedene Ergebnisse deklariert.

Es gibt auch einen Bericht [4] von Anleitungen, die Empfehlen einen User-Agent Switcher zu nutzen, um Einbrucherkennungssysteme zu umgehen, die versuchen Nutzer wiederzuerkennen. Dabei wird der Fingerprint verändert und kann auch Inkonsistenzen haben, ein naiver Fingerprintingalgorithmus wird den Fingerprint als neu und einzigartig erkennen. Aus diesem Grund wird auch vorgeschlagen, bei solchen Anwendungszwecken auf schlechter fälschbare Merkmale zu achten.

2.6.7 Entzug der Kommunikation

Da Browserfingerprinting auf der Analyse von Kommunikation basiert, kann es verhindert werden, indem der analysierenden Partei die Kommunikation entzogen wird. Dies bedeutet, dass auch ein eventueller Mehrwert, der durch diese Kommunikation entsteht, auch verloren geht.

Für bestimmte Formen von Browserfingerprinting kann der analysierenden Partei die Kommunikation entzogen werden. Ist die analysierende Partei eine Drittpartei, kann der Zugriff auf die Kommunikation über verschlüsselte Verbindungen verhindert werden.

Ein anderer Fall, bei dem die Kommunikation entzogen werden kann, sind Trackingdienstleister, deren Analysecode als verstecktes Element in Webseiten von Drittparteien eingebunden wird [30]. Um die Analyse durch diese Dienstleister komplett zu unterbinden, wurden Plugins wie Ghostery [2] entwickelt, die diese eingebundenen Elemente erkennen und entfernen. Dadurch wird nicht mit dem Trackingdienstleister kommuniziert und dieser kann keine Analyse durchführen.

2.6.8 Fälschung von Fingerprints

Werden Fingerprints genutzt, um Sessions abzusichern oder Accounts an Nutzer zu binden, muss ein Angreifer bestimmte Fingerprints nachstellen. Dazu muss er Kenntnis über den zu fälschenden Fingerprint erlangen. Gelingt ihm dies, muss er noch den Fingerprint nachahmen.

Ein Mittel, um einen Fingerprint nachzuahmen, ist das Verändern von Browsereinstellungen. Normalerweise nicht veränderbare Einstellung, wie der Useragent, können mit Plugins wie dem User Agent Switcher geändert werden [22]. Falschinformationen können allerdings über Inkonsistenzen in den Angaben [36] oder über kaum veränderbare Eigenheiten im Syntaxparsing des Browsers [6] erkannt werden. Aber auch diese Analysen können aber wiederum getäuscht werden, indem die Installation des Opfers komplett nachgebildet wird. Je nach Intensität des Fingerprints könnte dies aber auch das der Nachbildung des Betriebssystems und der Hardware bedürfen.

2.7 Zusammenfassung

Das Browserfingerprinting ist eine gut abgesicherte Methode, Browserinstallationen und ihre Nutzer wiederzuerkennen. Die Methode wurde theoretisch modelliert und mehrfach empirisch bewiesen. Es ist bekannt, dass diese Technik in der Industrie genutzt wird und es ist bekannt, welche Eigenheiten oft für das Browserfingerprinting genutzt werden.

Die Nutzungsmöglichkeiten des Browserfingerprintings können dem Interesse der Nutzer sowohl widersprechen als auch zuträglich sein. Die Methoden, die dem Nutzerinteresse eher widersprechen, da sie ihn gegen seinen Willen identifizieren oder beschränken, benötigen dafür ein sehr hohes Maß an Selbstinformation. Die Nutzungsmöglichkeiten, die vorhandene Sicherheitsmechanismen ergänzen, funktionieren mit viel Information besser, aber auch schon mit wenig Information.

Es gibt eine Reihe bekannter Ansätze, um Browserfingerprinting zu verhindern. Diese sind bei Weitem nicht so gut erforscht wie das Browserfingerprinting selbst. Viele Gegenmaßnahmen schränken den Nutzer teilweise stark ein oder sind nicht umgesetzt. Ein eindeutiger Favorit unter den Gegenmaßnahmen ist nicht bekannt.

MODELL

In der Studie „panopticlick“ von Eckersley [17] wurde ein mathematisches Modell des Fingerprintingprozesses genutzt, um Aussagen über die Einsetzbarkeit des Browserfingerprintings zu treffen. Auf diesem Modell basiert unter anderem auch diese Arbeit.

Bei diesem Modell wird die als Fingerprint gesammelte Information als Ergebnis eines Zufallsexperiments interpretiert. Der Inhalt der gesammelten Informationen spielt dabei keine Rolle, da Aussagen über das Zufallsexperiment getroffen werden sollen. Das Zufallsexperiment ist dabei die Installation, Konfiguration und Nutzung eines Browsers. Gelingt es, Informationen über die Zufallsverteilung zu erlangen, kann dadurch auf Eigenschaften der gesammelten Fingerprints geschlossen werden.

Die Messung eines Fingerprints mit einem Fingerprintingalgorithmus entspricht dabei der Funktion $F(\cdot)$ und der gemessene Fingerprint für eine Browserinstallation x entspricht $F(x)$. Die Fingerprints $f_n, n \in [0, 1, \dots, N]$ treten dabei mit der Wahrscheinlichkeit $P(f_n)$ auf. Die Zufallsverteilung und sogar die Anzahl der Fingerprints N ist dabei eine Unbekannte. Der Informationsgehalt I , also die Menge der zur Unterscheidung nutzbaren Information, eines Fingerprints f_n , ist durch die Formel $I(F(x) = f_n) = -\log_2(P(f_n))$ gegeben. Die Entropie der Zufallsverteilung entspricht dem Erwartungswert der preisgegeben Information für ein zufälliges f_n und hat die Formel: $H(F) = -\sum_{n=0}^N P(f) \log_2(P(f_n))$

Um die Messung eines Fingerprints detaillierter analysieren zu können, ist es wünschenswert, Teilmessungen und nicht nur die gesamte Messung als Block betrachten zu können. Um dies zu erreichen, können die Teilmessungen als Messungen mit verschiedenen Algorithmen $F_n(\cdot), n \in [0, 1, \dots, M]$ interpretiert werden. Da die einzelnen Messungen wie der Useragent und der Browsertyp voneinander abhängig sein können, können diese aber nicht einfach kombiniert werden. Sind die Abhängigkeiten zwischen Messungen mit den Fingerprintingalgorithmen $F_s(\cdot)$ und $F_t(\cdot)$ bekannt, kann die Menge, der insgesamt preisgegeben Information, mit der Formel $I_{s+t}(f_{n,s}, f_{n,t}) = -\log_2(P(f_{n,s}|f_{n,t}))$ bestimmt werden. Dass die Abhängigkeiten zwischen Messungen bekannt sind, kann allerdings nicht allgemein angenommen werden.

ANMERKUNG: mit der Formel stimmt was nicht oder ich hab was falsch verstanden. Ich würde sagen es muss $I_{s+t}(f_{n,s}, f_{n,t}) = -\log_2(P(f_{n,s}|f_{n,t}) * P(f_{n,t}))$

Dieses Modell wurde von Eckersley genutzt, um mittels einer Messung vieler Browserfingerprints und eines

konkreten Fingerprintingalgorithmus $F_{pan}(\cdot)$ auf Eigenschaften der Wahrscheinlichkeitsverteilung $P(f_n)$ zu schließen. Auf diese Weise konnte zu der gemessenen Probe eine Untergrenze der Entropie angegeben werden und Abschätzungen über die Menge an Informationen getroffen werden, die eine Browserinstallation oder ein Merkmal preisgibt.

Auf Basis dieses Modells wird die Grenze von 33 Bit Entropie genannt, die überschritten werden müsste, um alle Menschen dieses Planeten zu identifizieren [42]. Dies beruht darauf, dass $2^{33} = 8.589.934.592 > 7,2 \text{ Milliarden} \approx \text{AnzahlMenschen}$. Es kann aber nicht immer davon ausgegangen werden, dass die Menge der Nutzer der Menge der Menschen auf diesem Planeten entspricht. Sollen tausend Nutzer oder sogar nur 2 Nutzer voneinander unterschieden werden, reichen im Bezug auf diese Nutzergruppe bereits 10 Bit Entropie aus, um die Browserinstallation zu identifizieren, da $2^{10} = 1024 > 1000$.

Eine weitere Betrachtungsweise, die aus der Perspektive eines einzelnen Nutzers hilfreich ist, ist das Anonymity-Set. Ein Anonymity-Set ist die Menge der Browserinstallationen, die einen identischen Fingerprint beziehungsweise identische Merkmale haben. Hat ein Browser ein Anonymity-Set, das größer als 1 ist, gibt es mindestens einen anderen Browser mit identischen Merkmalen. Er kann also nicht von diesem unterschieden, also auch nicht eindeutig identifiziert werden. Wird trotzdem eine Identifizierung versucht, gelingt diese nach einfacher Rechnung mit der Wahrscheinlichkeit $\frac{1}{|\text{Anonymityset}|}$.

GEGENMASSNAHMEN

Um Nutzerverfolgung mittels Browserfingerprinting zu verhindern oder zu behindern, gibt es bereits einige Ansätze, die in Kapitel 2.6 vorgestellt wurden. Diese wurden aber nur teilweise auf ihre Effektivität hin überprüft oder sind noch unerforscht.

Um auch neue Methoden entwickeln und besser Gegenmaßnahmen für die weitere Untersuchung auswählen zu können, werden in Kapitel 4.1 die Stärken und Schwächen des Browserfingerprintings diskutiert. Anschließend werden die Gegenmaßnahmen und Strategien in Kapitel 4.2 genauer dargestellt, die in dieser Arbeit untersucht werden, und in Kapitel 4.3 Gegenmaßnahmen und Strategien dargestellt, die nicht weiter betrachtet werden.

4.1 Stärken und Schwächen

Das Browserfingerprinting hat prinzipielle Stärken und Schwächen, die von dem eingesetzten Fingerprinting-Algorithmus unabhängig sind oder für ganze Gruppen von Fingerprinting-Algorithmen gelten, diese werden hier möglichst vollständig aufgelistet. Die Stärken und Schwächen werden unter dem Fokus der Nutzerverfolgung betrachtet und als Orientierung genutzt, um die zu überprüfenden Gegenmethoden zu bewerten.

4.1.1 Falsch negative Ergebnisse

Einer der Fehler, die bei dem Browserfingerprinting auftreten können, ist das Versagen, Browserinstallationen wiederzuerkennen. Tritt dieser Fehler auf, kann dies bedeuten, dass die Verfolgung eines Nutzers unterbrochen wird. Die zu dem alten und neuen Fingerprint angesammelten Daten könnten also nicht mehr in Verbindung gebracht werden und der nutzbare Datensatz wäre geringer.

Tritt dieser Fehler nur bei bestimmten Nutzern oder selten auf, beschränkt sich der Schaden auch auf diese Nutzer oder seltenen Ereignisse. Diese Fehler müssten also häufig bei einer großen Masse von Nutzern auftreten, um die allgemeine Nutzerverfolgung über Browserfingerprinting ernsthaft zu behindern.

4.1.2 Falsch positive Ergebnisse

Der zweite grundsätzliche Art, wie Browserfingerprinting versagen kann, ist die Erkennung unterschiedlicher Browserinstallationen als eine identische. Wird dieser Fehler nicht abgefangen, würden die Aktivitäten mehrerer Nutzer zusammengefasst und als eine Gesamtheit bewertet werden.

Falsch positive Ergebnisse sind für das Browserfingerprinting also wesentlich problematischer als falsch negative, da nicht nur Daten verloren gehen, sondern fehlerhafte Verbindungen zwischen den Daten der beteiligten Nutzer und Falschinformationen erzeugt werden.

4.1.3 Informationsnutzung

Einige Informationen, die ins Browserfingerprinting einbezogenen werden, sind für das Aufbau von Webseiten bedeutend. Informationen wie die gewünschte Sprache, die unterstützten Komprimierungsarten oder der Useragent werden von manchen Webseiten genutzt, um auf Nutzergruppen angepasste Webseiten zu generieren. Ist dies der Fall, können diese Informationen nicht beliebig verändert werden, ohne die Benutzbarkeit des Browsers einzuschränken.

Ist eine große Masse von Nutzern oder eine große Menge von Webseiten betroffen, ist es schwer einzuschätzen, welche Merkmale eines Browsers genutzt werden, um Webseiten auf Nutzer anzupassen. Dies bedeutet, dass Manipulationen von Merkmalen aufgrund zentral gegebener Regeln das Risiko bergen, die Benutzbarkeit des Browsers einzuschränken.

4.1.4 Nutzung zusätzlicher Informationen

Stehen Informationen über den Browserfingerprint hinaus zur Verfügung, können diese mit dem Fingerprint kombiniert, werden um den Nutzer wiederzuerkennen.

Dies erleichtert die Verfolgung der Nutzer sehr, da durch diese Kombination die Menge der preisgegeben Information sinkt, die notwendig ist, um einen Nutzer wiederzuerkennen. Die kann wie beim Device Fingerprinting, das Einbeziehen des TCP/IP-Stacks [42] sein oder die IP-Adresse sein [48].

4.1.5 Seitenkanäle

Seitenkanäle sind bei verschiedener Software ein Problem, da sie unerwünscht Informationen freigeben. Dieses Problem kann zwar auf verschiedene Weisen angegangen werden, bleibt aber bestehen. Die Seitenkanäle erlauben, selbst bei kritischen Anwendungen wie Verschlüsselung [19], geheime Informationen auszulesen.

Seitenkanäle werden auch beim Browserfingerprinting genutzt, um Informationen zu gewinnen. In Anbetracht der Probleme bei anderen Anwendungen und der Komplexität von Browsern ist es schwer vorstellbar, dass diese Seitenkanäle in Browsern eliminiert werden können.

4.1.6 Statistische Abhängigkeiten

Eine detaillierte Analyse der bereitgestellten Funktionen, um die Browserversion zu ermitteln, bringt kaum etwas, wenn der Useragent diese Information bereits in sich trägt. Die Abhängigkeiten zwischen den einzelnen gemessenen Merkmalen haben sowohl Nachteile als auch Vorteile für das Fingerprinten von Browsern. Als Beispiel sollen hier ein bestimmter Satz von bereitgestellten Funktionen $A_{firefox}$ und der Useragent $B_{firefox}$ gemessen werden, wobei $P(A_{firefox}|B_{firefox}) \approx 1$.

Dies arbeitet einerseits gegen das Browserfingerprinting, da sich die Gesamtmenge der preisgegebenen Information durch das Erkennen von $A_{firefox}$ im Wissen von $B_{firefox}$ nicht oder nur kaum vergrößert. Es

würde lediglich die Information gewonnen, dass der Useragent nicht auf bestimmte Arten gefälscht wurde. Von dieser Information ist aber zu kein hohes Maß an Unterscheidbarkeit zu erwarten.

Andererseits bieten diese Abhängigkeiten die Möglichkeit Fälschungen zu erkennen, denn in diesem Fall ist beispielsweise $P(A_{IE}|B_{Firefox})$ und nicht $P(A_{Firefox}|B_{Firefox})$ relevant. Dabei muss $P(A_{IE}|B_{Firefox}) \leq 1 - P(A_{Firefox}|B_{Firefox})$ sein, wobei $P(A_{Firefox}|B_{Firefox}) \approx 1$ weiterhin gilt. Dadurch ist $I_{P(A_{IE}|B_{Firefox})} = -\log_2(1 - P(A_{Firefox}|B_{Firefox})) \approx -\log_2(0) = \infty$. Als Beispiel könnte ausschließlich der Useragent gefälscht sein und detailliertere Angaben dem Useragent widersprechen. Dadurch wird klar, dass der Useragent gefälscht ist, wovon ein recht hohes Maß an Unterscheidbarkeit zu erwarten ist. Zusätzlich dazu ist auch die Art und Weise der Fälschung bekannt, wodurch sich die Fälscher noch einmal untereinander differenzieren.

Diese Effekte treten auch bei einfachen Auswertungslogiken für Fingerprints wie Tests auf Gleichheit ein, solange $A_{Firefox}$, A_{IE} und $B_{Firefox}$ gemessen werden.

4.1.7 Datenübermittlung

Bei auf Skripten basierendem Fingerprinting werden Merkmale des Browsers bestimmt. Dies geschieht zunächst nicht auf dem Server, sondern im Browser selbst. Die ermittelten Daten müssen nun dem Analyseserver aber auf irgendeine Art und Weise mitgeteilt werden. Scheitert diese Übermittlung der Daten, können die ermittelten Merkmale nicht verwendet werden.

Auch andere Arten des Browserfingerprintings benötigen Informationen, die vom Browser preisgegeben werden und einem Analyseserver mitgeteilt werden müssen. Wird die Übermittlung dieser Informationen abgefangen, können auch diese Informationen nicht verwertet werden. Das Browserfingerprinting ist also prinzipiell dagegen anfällig, dass die Kommunikation des Browsers manipuliert wird.

4.1.8 Verknüpfbarkeit von Fingerprints

Eines der Probleme des Browserfingerprintings, das schon früh erkannt wurde, ist die Stabilität der Browserfingerprints. Instabil sind Browserfingerprints dann, wenn sie sich ändern. Dies kann zum Beispiel durch Updates oder Neukonfigurationen geschehen. Sind die Fingerprints instabil, kann ein naiver Fingerprintingalgorithmus den Zusammenhang zwischen f_n und f_{neu} nicht herstellen. Also können diese nicht verknüpft werden und werden fehlerhaft als zwei verschiedene Browserinstallationen erkannt.

In den Arbeiten von Eckersley [17] und Tillmann [42] zum Browserfingerprinting wurden Algorithmen demonstriert, die die Veränderungen der meisten Fingerprints ausgleichen konnten. Eckersley sieht aber in dieser Instabilität, eine eventuelle Möglichkeit gegen Browserfingerprints vorzugehen. Die Beweise und Formeln der Kernteile der Arbeiten von Eckersley und Tillmann beschäftigen sich nicht weiter mit dieser Erweiterung des Browserfingerprintings, wodurch die Beweise und Formeln dieser Arbeiten nicht zwangsläufig gelten, wenn solche Algorithmen eingesetzt werden.

Sind einzelne Merkmale sehr instabil, ist nicht klar, ob die Nutzerverfolgung durch ihre Einbeziehung profitieren würde. Einerseits stehen zwar mehr zur Unterscheidung nutzbare Informationen zur Verfügung und der Fingerprint wird eindeutiger. Andererseits vergrößert sich damit auch die Instabilität des Fingerprints und es müssen mehr Änderungen ausgeglichen werden.

4.1.9 Vorgangscharakter

In den zu Browserfingerprinting existierenden Arbeiten wird das Fingerprinten eines Browsers als einmalige Messung behandelt. Werden aber mehrere Messungen des Fingerprints zum Beispiel über Cookies oder Login-Vorgänge in Verbindung gebracht werden, können Veränderungen am Browserfingerprint erkannt und analysiert werden. Bei einem stabilen Fingerprint muss das Ergebnis einer solchen Analyse lediglich sein, dass sich der Fingerprint nicht geändert hat.

Verändert sich ein Fingerprint aber, kann diese Veränderung auf Muster analysiert und in eine abstraktere Form des Fingerprints miteinbezogen werden. Der Fingerprintingalgorithmus könnte versuchen, Plugins zu erkennen, die die Merkmale von Browsern randomisieren, und darauf reagieren. Zusätzlich könnte es möglich sein, Merkmale des Browsers zum Fingerprinten zu nutzen, die über mehrere Zugriffe hinweg gemessen werden müssen. Als ein solches Merkmal könnte das Verhalten der 3,85 % der Bing Nutzer dienen, deren Cookies bei jeder einzelnen Anfrage erneut gelöscht werden [48]. Alleine durch diese Form des Cookielöschens werden $-\log_2(3,85\%) \approx 4,7$ Bit zur Unterscheidung nutzbarer Informationen preisgegeben.

4.1.10 Clientseitiger Code

Für aktives Browserfingerprinting muss dem Browser der Teil des Codes zur Verfügung gestellt werden, mit dem der Fingerprint erhoben wird. Dadurch ist eine Codeanalyse möglich, die es theoretisch erlaubt, das aktive Browserfingerprinting zu erkennen und weiter zu analysieren. Die dabei genutzten Schwächen und Fingerprintingmethoden können so erkannt und effektiver bekämpft werden. Eine solche Analyse kann allerdings sehr aufwendig sein, wenn die Fingerprintingskripte, wie bereits teilweise praktiziert [9], Analysen behindern.

Auch bestehen für clientseitigen Code keine Garantien für die korrekte Ausführung des Codes. Der Code kann fehlerhaft, manipuliert oder auch gar nicht ausgeführt werden. Dies erlaubt zum Beispiel das Nutzen von manipulierten Browsern, um Browserfingerprintingskripte zu suchen.

Dieser Nachteil für Browserfingerprinting wird dadurch abgemildert, dass eine automatische Analyse nicht immer möglich ist. Einem automatischen Erkennen aller Browserfingerprintingskripte widerspricht die Unmöglichkeit, das Halteproblem zu lösen. Aus diesem lässt sich folgern, dass die Funktionsweise von Programmen nicht allgemein bestimmbar ist. Daraus folgt letztendlich eine Situation, wie sie aus dem Antivirus-Bereich und böartigen Skripten bekannt ist.

4.1.11 Serverseitiger Code

Verschiedene Teile des Browserfingerprinting-Prozesses laufen auf einem Analysesystem ab. Dies betrifft das passive Erheben von Browsermerkmalen, das Speichern der Fingerprintdatenbank und die Nutzung dieser Datenbank. Solange das Analysesystem selbst nicht untersucht werden kann, können diese Vorgänge nicht direkt analysiert werden und nur Vermutungen über sie angestellt werden.

Dies ist ein großer Vorteil für das Browserfingerprinting, da manche Typen wie passives Browserfingerprinting von den Nutzern nicht erkannt werden können. Dass die Art der Speicherung und die genaue Weise des Vergleichs von Fingerprints nicht direkt beobachtbar ist, erschwert zudem eine Untersuchung von eingesetzten Fingerprintingalgorithmen.

4.1.12 Kommunikation

Das Browserfingerprinting basiert auf dem Besuch einer Webseite mit einem Browser. Fände dieser Besuch nicht statt, könnte kein Fingerprint erhoben werden und das Browserfingerprinting wäre unmöglich.

Dies ist für das Browserfingerprinting sowohl von Vorteil wie von Nachteil, da dieser Webseitenbesuch für den Nutzer nicht gewinnbringend sein muss, es aber sein kann. Bei Webseiten, die keinen Vorteil für den Nutzer bieten, ist ein Boykott dieser Seiten und somit das Verhindern des Browserfingerprintings denkbar. Bietet die Webseite Vorteile für den Nutzer oder wird versucht einen Besuch auf dieser Webseite zu erzwingen, wird Boykott wahrscheinlich nicht stattfinden.

4.1.13 Nutzung der Fingerprints

Werden Browserprints erhoben, ist anzunehmen, dass diese Browserfingerprints auch verwendet werden. Dies hat zwei Nachteile für das Browserfingerprinting.

Erstens kann versucht werden, von der Nutzung der Fingerprints auf nicht sichtbare Teile eines Fingerprintingalgorithmus zu schließen. Dies kann beispielsweise in Form von Orakeln passieren, die verraten, ob ein Fingerprintingalgorithmus getäuscht werden konnte oder nicht [37].

Zweitens ist anzunehmen, dass die Nutzung der Fingerprints Ansprüche an die Güte der Fingerprints hat. Treten zu viele Fehler auf, wäre es vorstellbar, dass die Gütekriterien für eine weitere Verwendung unterschritten werden und eine weitere Verwendung verunmöglicht wird.

4.1.14 Manipulation des Zufallsprozesses

Das Modell des Browserfingerprinting geht davon aus, dass die Installation und Konfiguration ein Zufallsprozess mit Abhängigkeiten zwischen den Messungen ist. Da aber beständig neue Browser unter neuen Bedingungen hinzugefügt werden und alte Browserinstallationen verschwinden, ist dieser Zufallsprozess nicht statisch. Auch bestehen Abhängigkeiten von Messungen des Zufallsprozesses und der Neukonfiguration oder Abänderungen von Browsern. Dies ist zum Beispiel der Fall, wenn Merkmale verändert werden, weil sich in einer Messung herausgestellt hat, dass sie auffällig sind.

Durch Änderungen an Browsern kann also der Zufallsprozess selbst verändert werden. Werden Änderungen an vielen Browsern oder koordiniert vorgenommen, sind auch bewusste Manipulation des Zufallsprozesses denkbar. Dadurch könnte die Verlässlichkeit des Browserfingerprints verringert werden, indem die Stabilität der Browserfingerprints oder die Entropie der Zufallsverteilung reduziert wird.

4.1.15 Schwacher Beweis der Funktionsweise

Das Browserfingerprinting ist gut erforscht und viele Nutzer sind mit dieser Technik identifizierbar. Dies ist allerdings kein Beweis dafür, dass Browserfingerprinting für alle Nutzer und auch in der Zukunft funktioniert. Studien können nur Aussagen über den Zeitraum der Studie und Prognosen über die Zukunft treffen. Für exakte Aussagen über die Zukunft des Browserfingerprintings ist dieses zu sehr von konkreten Fingerprintingmethoden, eingesetzten Browsern und der Gefahrenwahrnehmung der Nutzer abhängig.

Die Abschätzung der Verteilung der Fingerprints wird auch dadurch erschwert, dass die dafür genutzten Fingerprints nicht über lange Zeiträume hinweg gesammelt werden dürfen. Werden Browserfingerprints über lange Zeit gesammelt wie bei dem seit 2010 zugänglichen „panopticlick“, wird die aufgebaute Zufallsverteilung dadurch verzerrt, dass alte und neue Fingerprints vermischt werden. Selbst unter der Annahme, dass lediglich 75 % der gesammelten Fingerprints heutzutage nicht mehr relevant sind, wird die preisgegebene Information von Fingerprints bei „panopticlick“ bereits um 2 Bit überschätzt, da $I = -\log_2\left(\frac{|anonymityset|}{|fingerprints|}\right) = -\log_2\left(\frac{|anonymityset|}{|fingerprints|} * 4\right) = -\log_2\left(\frac{|anonymityset|}{|fingerprints|}\right) - \log_2(4) = -\log_2\left(\frac{|anonymityset|}{|fingerprints|}\right) - 2$.

Es gibt auch keinen Grund dafür, warum nicht mehrere Nutzer durch Zufall denselben Fingerprint haben sollten und in den Studien zu Browserfingerprinting sind Nutzer, die sich einen Fingerprint teilen, üblich. Dies sorgt dafür, dass bei Erkennen eines identischen Fingerprints nicht ohne Restzweifel gefolgert werden kann, dass dieselbe Browserinstallation vorliegt. Aufgrund dessen muss davon ausgegangen werden, dass ein Browserfingerprintingalgorithmus ein gewisses Maß an Fehlern produziert, die in irgendeiner Weise toleriert werden müssen.

4.2 Untersuchte Gegenmaßnahmen und Strategien

In diesem Abschnitt werden die Gegenmaßnahmen und Strategien aufgelistet, die in der weiteren Arbeit untersucht werden. Diese Untersuchung findet in Kapitel 5 und Kapitel 6 statt.

4.2.1 Randomisieren von Fingerprints

Ein intuitiver Ansatz ist das Randomisieren von Browsermerkmalen. Dadurch soll der Browserfingerprint so instabil werden, dass er nach einer Änderung nicht mehr wiedererkannt werden kann. Funktioniert dies, wird der Nutzer trotz einzigartigem Fingerprint nicht wiedererkannt.

Dies wurde mit dem Plugin PriVaricator [37] bereits untersucht, welches das Implementieren einer Chromiumvariante beinhaltet, dass gängige Fingerprintingalgorithmen auf diese Weise überlisten konnte. Aufgrund von Bedenken bezüglich der Benutzbarkeit des Browsers wurden dabei aber nur der Useragent und die Liste der installierten Schriften teilweise randomisiert.

Trotz des guten Forschungsstands wird diese Methode noch einmal untersucht, damit Effekte wie der Vorgangscharakter oder die Reaktionen auf die Randomisierung seitens der Browserfingerprintingskripte einbezogen werden können.

4.2.2 Automatische Browserupdates

Der Browsertyp und die Browserversion sind Merkmale, die große Abhängigkeiten zu anderen Merkmalen haben. Diese sind zum Beispiel die Javascriptengine oder die Menge der ohne Plugins zur Verfügung gestellte Funktionen. Wäre die Browserversion im Bezug auf den Browsertyp standardisiert, wären also auch die von der Browserversion abhängigen Merkmale standardisiert.

Über automatische Browserupdates könnte versucht werden, diese Standardisierung von Browserversionen zu erzwingen. Da dies auch noch andere Sicherheitsvorteile bietet, haben manche Browser bereits automatische oder sogar erzwungene Updates. Der Vorteil, den einem solches System bieten könnte, ist aus Abbildung 4.1 erkennbar. Hier ist deutlich zu erkennen, dass die eingesetzten Firefoxversionen deutlich konsistenter als die Internetexplorerversionen sind, obwohl die Firefoxversionen instabiler sind als die Internetexplorerversionen.

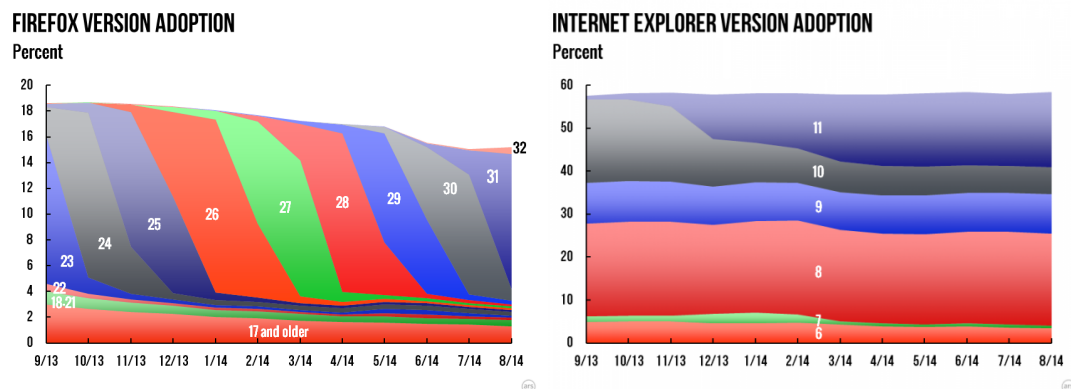


Abbildung 4.1: Visualisierung der Versionsadaption Firefox und Internetexplorer über den Verlauf eines Jahres.

Bilder von arstechnica.com

ANMERKUNG: Die Bilder sind von einer Webseite. Gibt das Copyrightprobleme?

Da der Effekt eines solchen Mechanismus auf das Browserfingerprinting nicht untersucht wurde und die in Abbildung 4.1 genutzten Daten nicht wissenschaftlich abgesichert sind, soll dies in dieser Arbeit weiter untersucht werden.

4.2.3 Deaktivieren von Skriptsprachen

Die wohl am meisten empfohlene Gegenmaßnahme gegen Browserfingerprinting ist das Deaktivieren von Skriptsprachen über Browserplugins wie NoScript. Da Browserfingerprinting viel Informationen durch die

Nutzung von Skripten gewinnen kann, besteht die Vermutung, dass dieses durch das Deaktivieren von Javascript behindert wird.

Werden spezialisierte Plugins genutzt, um Skripte zu deaktivieren, ist es normalerweise möglich, einzelne Skripte temporär oder permanent zu aktivieren beziehungsweise zu deaktivieren. Dadurch soll es möglich sein, Webseiten zu nutzen, auch wenn diese Javascript zur Darstellung benötigen. Dies ist ein gängiges Problem bei dem Deaktivieren von Javascript.

Das Deaktivieren von Javascript mit solchen Plugins wird in dieser Arbeit untersucht, da dieses oft empfohlen wird, aber kaum wissenschaftlich untersucht wurde.

4.2.4 Fälschung von beliebigen Fingerprints

Wird Browserfingerprinting genutzt, um zum Beispiel Sessions abzusichern, kann es notwendig sein, einen bestimmten Fingerprint zu fälschen, um dieses Session zu übernehmen. Es könnte aber auch versucht werden dies als Maßnahme gegen Browserfingerprinting zu nutzen, indem ein Browserfingerprint mit einer möglichst geringen Menge an preisgegebenen Informationen gefälscht wird. Als Methoden einen Fingerprint mit einer möglichst geringen Menge an preisgegebener Information zu finden, sollen bekannte Fingerprints und Messungen von Fingerprints genutzt werden.

Dies wird in dieser Arbeit mit Hinblick auf die technische Komplexität einer Fälschung und der Effektivität einer solchen Fälschung untersucht.

4.2.5 Koordiniertes Fälschen von Fingerprints

Um die erwarteten Probleme beim Fälschen von beliebigen Fingerprints abzumildern, kann versucht werden nicht beliebige, sondern einfach zu fälschende Fingerprints zu fälschen. Dazu sollen Merkmale nicht verändert werden von denen eine große Menge von Attributen abhängt. Merkmale von denen keine anderen Merkmale abhängen, sollen bevorzugt verändert werden.

Um dies zu untersuchen soll ein Browserplugin angenommen werden, dass versucht Fingerprints koordiniert zu fälschen. Dieses Plugin soll ein Menge von Merkmalen kennen, von denen es ausgeht, dass sie gefahrlos fälschbar sind. Um einen Kandidat zum Fälschen von Fingerprints zu finden soll ein Vermittlungsserver angefragt und änderbare Merkmale übergeben werden. Diese Vermittlungsserver bestimmt nun den Fingerprint des Browsers und gibt eine Menge von Fingerprints zurück, die genutzt werden und gefahrlos fälschbar sind. Das Plugin soll anschließend einen der vorgeschlagenen Fingerprints fälschen. Die Datenbank der Fingerprints des Vermittlungsservers soll über das Fingerprinten von Nutzern des Plugins und Internetnutzern aufgebaut werden.

4.2.6 Blockieren von Kommunikation

Da von erfolgreichen Blockieren von Kommunikation eine sicheres Verhindern von Browserfingerprinting zu erwarten ist, wird dieses in dieser Arbeit betrachtet. Dazu wird die Effektivität vom Blockieren von Kommunikation zu bestimmten IPs untersucht werden, wie es über eine Firewall realisiert werden kann. Das Blockieren der IPs soll anhand einer zentral betreuten Sammlung von IPs geschehen, in der IPs aufgenommen werden, deren einzige Funktion das Browserfingerprinting ist. Die IPs sollen durch Besuchen von Webseiten mit manipulierten Browsern und Hinweisen aus der Öffentlichkeit aufgebaut werden.

4.2.7 Filtern von Kommunikation

Muss mit einem Server kommuniziert werden, kann versucht werden die Kommunikation zu diesem Server zu filtern. Das Browserfingerprinting ist zwar möglich, es kann aber so versucht werden Browserfingerprinting und insbesondere aktives Browserfingerprinting einzuschränken. Gefiltert werden kann der Analysecode, der vom Server zum Browser gesendet wird, oder die über den Browser erhobenen Daten, die vom

Browser zum Server geschickt werden. Da die populären Plugins „AddBlock“ und „Ghostery“ angekündigt haben, neben ihrem eigentlichen Einsatzzweck auch Browserfingerprintingskripte zu blocken, wird dies in dieser Arbeit untersucht.

Die Regeln, die zum Filtern der Kommunikation genutzt werden, sollen von einer zentral betreuten Datenbank geladen werden. Die Datenbank soll durch Analyse von Browserfingerprintingskripten und Hinweise aus der Öffentlichkeit aufgebaut werden. Browserfingerprintingskripte sollen mit einem manipulierten Browser gefunden werden.

4.2.8 Kombination verschiedener Maßnahmen

Um die Schwächen von Maßnahmen gegen Browserfingerprinting auszugleichen kann versucht werden, mehrere Gegenmaßnahmen miteinander zu kombinieren. In Kombination könnten so Angriffe vermieden und zusätzliche Ebenen des Schutzes geboten werden, um bei Umgehung einer Schutzmaßnahme trotzdem eine Wiederidentifizierung verhindern zu können. Auch der „Privacy Paradox“ könnte auf diese Weise umgangen werden, da die Nutzer der verschiedenen Maßnahmen nicht auf verschiedene Plugins mit unterschiedlichem Verhalten angewiesen sind.

Die Formulierung einer solchen Kombination findet erst nach der Untersuchung der einzelnen Maßnahmen statt, um eine sinnvolle Kombination von Maßnahmen bestimmen zu können.

4.3 Nicht untersuchte Gegenmaßnahmen und Strategien

Die Gegenmaßnahmen, die in dieser Arbeit bewusst nicht untersucht werden, werden hier der Vollständigkeit halber trotzdem kurz dargestellt.

4.3.1 Illegalisierung von Browserfingerprinting

Um Browserfingerprinting zur Nutzerverfolgung zu behindern, kann es Verboten oder dessen Nutzung teilweise illegalisiert werden. Um eine anderweitige Nutzung von Browserfingerprinting wie die Absicherung von Sessions nicht einzuschränken, sollte das Verbot nur die Nutzerverfolgung betreffen oder andere Nutzungen mit Ausnahmen zu versehen.

Als eine Basis für eine solche rechtliche Konstruktion könnte die Regelung der EU zu Cookies dienen, die Cookies und Pseudocookies nur mit Zustimmung des Nutzers erlaubt. Das Verbotes dürfte sich allerdings nicht mehr das Speichern von Daten im Browser beziehen, sondern einen Kernvorgang des Browserfingerprintings betreffen. Dies könnte das Erheben, Speichern oder Übermitteln von Daten über von Privatpersonen eingesetzten Computern und Programmen betreffen.

Als eine andere Basis für eine Illegalisierung von Browserfingerprinting können Datenschutzgesetze dienen. In diesen wird das Speicherung und die Weitergabe von personenbezogenen oder personenbeziehenden Daten bereits reguliert. Das könnte dahingehend erweitert werden, dass Daten, die in Bezug zu Merkmalen von Personen oder ihren technischen Systemen gespeichert werden, nicht als pseudonym, sondern als personenbeziehbar gewertet werden.

Eine Formulierung und Bewertung eines solchen Vorschlages ist allerdings etwas für einen Juristen. Absehbar ist allerdings, dass politische und juristische Probleme überwunden werden müssen, um den rechtlichen Rahmen zu verändern. Auch wenn dieser rechtliche Rahmen verändert wird, sind Durchsetzungsschwierigkeiten zu erwarten.

ANMERKUNG: Hier wird noch ein konkreter Bezug zu der EU Datenschutzrichtlinie eingebaut

4.3.2 Aufklärung der Nutzer

Das Browserfingerprinting betrifft eine breite Masse an Nutzern. Diese Nutzer sind sich der Existenz und der Funktionsweise des Browserfingerprintings nicht unbedingt bewusst. Durch Aufklärung der Nutzer kann versucht werden politischen Druck aufzubauen und eine Nutzerbasis zu schaffen, die an Gegenmaßnahmen teilnimmt. Eine solche Nutzerbasis kann zum Beispiel helfen Browserfingerprinting weiter zu erforschen oder das „privacy paradox“ zu überwinden.

Um eine solche Aufklärung zu erreichen, könnte das Browserfingerprinting, begleitet von Medienarbeit, weiter erforscht werden. So würde die Bevölkerung aufgeklärt und das Browserfingerprinting greifbarer.

Dieser Ansatz ist zwar vielversprechend, wird aber in dieser technisch ausgerichteten Arbeit nur kurz behandelt. Wissenschaftliche Veröffentlichung zum Thema Browserfingerprinting haben mit mehreren Zeitungsartikeln in großen Zeitungen bereits eine gute Medienresonanz gehabt. So wurden wie in Appendix XX in einer unvollständigen Suche zu Eckersley Studie 11 Berichte, zu Tillmanns Diplomarbeit 10 Berichte und zu der neuesten Studie zu Canvas Fingerprinting 21 Berichte gefunden. Die tatsächliche Zahl der Berichte ist vermutlich größer, aber auch diese Zahlen zeigen bereits ein großes Interesse der Medien an diesem Thema. Ein weiterer Faktor ist, dass ein Großteil der Bevölkerung mit der Nutzerverfolgung im Internet nicht einverstanden ist [42] und somit für solche Artikel ansprechbar ist.

ANMERKUNG: Ich bin mir nicht sicher ob die Angabe der Webseiten so ok ist.

4.3.3 Merkmale verheimlichen

4.3.4 Fälschen von Kommunikation

4.3.5 Vertrauen in Fingerprintstabilität schwächen

4.3.6 Fingerprinterspezifische Aktionen

4.3.7 Fingerprintänderungen timen

4.3.8 Erkennung von genutzten Merkmalen

5

THEORETISCHE BETRACHTUNG

Zur theoretischen Betrachtung der Gegenmaßnahmen wird auf Eckerleys Modell zurückgegriffen, das in Kapitel ?? dargestellt wurde. Dazu wird zunächst in Abschnitt 5.1 das Modell um einige Formeln erweitert und anschließend werden die Gegenmaßnahmen untersucht.

5.1 Modell

size anonset + size Fingerprints + size Analyse

$$I + 1 - > |anonset|/2$$

$$I = -\log_2\left(\frac{|anonset|}{|fingerprints|}\right)$$

Optimierungsziele:

- große Wahrscheinlichkeit für Merkmale gut, kleine schlecht
- großes Anonymity set gut, kleines schlecht
- große Instabilität gut, kleine schlecht
- großes I schlecht, kleines I gut
- große Entropie schlecht, kleine gut

5.2 Randomisieren von Fingerprints

Werden Merkmale eines Browsers randomisiert, kann dies folgendermaßen modelliert werden. Der Browser hat Merkmale, die ein normales Level an Stabilität haben und in dieser Betrachtung als komplett stabil angenommen werden. Die für diese Merkmale gemessenen Werte eines Browsers entsprechen dem Ereignis A . Die randomisierten Merkmale haben ein sehr geringes Maß an Stabilität und werden deswegen mit B

getrennt betrachtet. Die für diese Merkmale gemessenen Werte eines Browsers entsprechen den Ereignissen B_1, B_2, \dots, B_N , wobei N die Anzahl der möglichen Ergebnisse des Randomisierungsprozesses ist und $B = \{B_1, B_2, \dots, B_N\}$. Ein gemessener Fingerprint ist also $\{A, B_n\}$.

Wird nur schwach randomisiert, ist es vorstellbar, dass sich die gemessenen Fingerprints wiederholen und der Nutzer so trotzdem unter mehreren Fingerprints verfolgt werden kann. Kann der Browserfingerprint in mehreren Zuständen der Randomisierung gemessen werden, kann versucht werden die Randomisierung zu erkennen und den Nutzer trotzdem zu identifizieren. Aber auch ohne Kenntnis, dass ein spezieller Fingerprint randomisiert ist, kann versucht werden eine Randomisierung auszugleichen. Dies wird in den Abschnitten 5.2.1, 5.2.2 und 5.2.3 untersucht und in Abschnitt 5.2.4 zusammengefasst.

5.2.1 Notwendiges Maß der Randomisierung

Damit ein Nutzer aufgrund nicht ausreichender Randomisierung wiedererkannt werden kann, muss ein Ereignis B_n bei zwei verschiedenen Gelegenheiten gemessen werden. Das Ereignis A bleibt konstant und muss deswegen nicht betrachtet werden.

Eine obere Grenze für die Gelegenheiten, bei denen ein neuer Fingerprint präsentiert werden kann, kann leicht gegeben werden. Da B höchstens N verschiedene Ereignisse enthält, ist selbst bei optimaler Ausnutzung der Ereignisse für höchstens N Gelegenheiten ein unverbrauchtes Ereignis verfügbar.

Um eine Schätzung des notwendigen Maßes zu gewinnen, die sich näher an vorhandenen Ansätzen orientiert, wird ein Spezialfall einer solchen Randomisierung betrachtet. Bei diesem wird die Wahrscheinlichkeit für die Ereignisse als gleich verteilt angenommen und bei jeder Gelegenheit ein neues Ereignis auswürfelt. Soll dabei kein Ereignis wiederholt werden entspricht dies einem Ziehen ohne Zurücklegen aus B . Bei n Gelegenheiten gelingt dies mit $P_{anon} = \frac{N!}{(N-n)!}$.

	$N = 10^3$	$N = 10^6$	$N = 10^9$	$N = 10^{12}$
$n = 10$	95,5861%	99,9955%	100,0000%	100,0000%
$n = 100$	0,5959%	99,5062%	99,9995%	100,0000%
$n = 1000$	0%	60,6733%	99,9501%	100,0000%
$n = 10000$	—	0%	95,1234%	99,9950%
$n = 100000$	—	0%	0,6737%	99,5013%

Tabelle 5.1: P_{anon} für verschiedene n und N in Prozent auf 4 Stellen nach dem Komma gerundet

Wie aus Tabelle 5.1 hervorgeht, kann schon bei kleinem N und großem n ein solcher Vorgang fast sicher ausgeschlossen werden. Dabei werden selbst geringfügige Fehler ignoriert und nur das vollständige Funktionieren der Randomisierung betrachtet. Der Browser „PriVaricator“ [37] kam bereits durch die Randomisierung von *offsetHeight*, *offsetWidth* und *getBoundingClientRect* auf mindestens 10^6 mögliche Ereignisse, wodurch zu erwarten ist, dass kein Fingerprint doppelt generiert wird.

5.2.2 Erkennung des Randomisierers

Mit Hilfe von Cookies oder Logins können mehrere randomisierte Fingerprints für einen Browser gemessen werden. Werden dadurch verschiedene Ereignisse aus B gemessen, kann eine Änderung des dazugehörigen Fingerprints erkannt werden. Sind diese Änderungen häufig und regelmäßig, kann ein Randomisierer vermutet werden.

Ein Fingerprintingskript kann mit dieser Erkenntnis versuchen, die Randomisierung auszugleichen. Die erste Möglichkeit ist, die als randomisiert erkannten Merkmale, also B , zu ignorieren. Der Haupteffekt des Randomisierers wäre ausgehebelt und die Frage nach der Einzigartigkeit des Nutzers wäre wieder relevant.

Zur Unterscheidung des Nutzers stünde also noch $\{A\}$ zur Verfügung. Anstatt der gemessenen Merkmale muss etwas anderes gespeichert werden. Dies könnte $NULL$ aber auch ein Flag B^* sein, der das Wechseln der Werte anzeigt. Dadurch stünden nun $\{A, B^*\}$ zur Unterscheidung des Nutzers zur Verfügung. Würden in

der Änderung von B^* sogar stabile Muster M erkannt werden, stünde nun $\{A, B^*, M\}$ zur Unterscheidung von Nutzern zur Verfügung.

Dem Fingerprintingalgorithmus stehen zur Analyse nur die Ergebnisse von Zufallsexperimenten zur Verfügung, wodurch bei Aussagen auf die Zufallsverteilung immer ein Restzweifel besteht. Daher kann beim Einsatz eines Randomisierers nicht mit absoluter Sicherheit gesagt werden, welche Merkmale sich verändern und in welchem Ausmaß dies geschieht.

Wird B^* erkannt und reicht $\{A, B^*, M\}$ zur Identifikation des Nutzers aus, bietet der Randomisierer keinen Schutz gegen Browserfingerprinting.

Im Falle des „PriVaricators“ [37] wird nicht versucht, die Änderungen zu verbergen und gegen Seitenkanalattacken zu schützen. Dadurch ist anzunehmen, dass durch Ignorieren von B kaum Information verloren gehen und A trotz Randomisierung ein für einen Fingerprint normales Maß an Information preisgibt. Zum „PriVaricator“ finden sich auch keine Anleitungen und keine Projektseite, wodurch die Vermutung naheliegt, dass nur sehr wenige Nutzer von „PriVaricator“ existieren. Wird eine Randomisierung über den „PriVaricator“ erkannt, hat dieses Merkmal also ein kleines Anonymity-Set und muss ein sehr hohes Maß an Information besitzen. Aber auch bei einem populären Plugin würde dies relevantes Maß an Information preisgegeben. Als zusätzliche Informationsquelle bietet der „PriVaricator“ die Möglichkeit, die Parameter für die Randomisierung über Konfigurationsfiles zu ändern. Kann diese Konfiguration als Merkmal ausgelesen werden, kann sie genutzt werden, um zwischen Browserinstallation zu unterscheiden und damit zur Identifizierung des Nutzers beitragen.

5.2.3 Ignorieren von Merkmalen

Ist der Einsatz und Funktionsweise von Randomisierern zum Beispiel durch Projektseiten bekannt, können die Browserfingerprintingalgorithmen auf dieses reagieren, ohne zu wissen, welche Fingerprints randomisiert wurden. Dazu können die randomisierten Merkmale ignoriert werden. Das muss nicht zwangsweise für alle Nutzer passieren, wenn Randomisierer nur für bestimmte Browser verfügbar sind oder die Randomisierung nur in Kombination mit anderen Merkmalen auftritt. Im Falle des „PriVaricators“ [37] wäre dies zum Beispiel das Erkennen des Chromium-Browsers auf dem „PriVaricator“ basiert.

In diesem Fall stehen die randomisierten Merkmale B nicht mehr zur Verfügung, wenn die ansonsten gemessenen Merkmale A implizieren, dass ein Randomisierer eingesetzt wird. Es stehen also weiterhin $-\log_2(P(A))$ Informationen zur Verfügung.

Aus der Perspektive eines einzelnen Nutzers ist also $-\log_2(P(A))$ und somit die Menge und Seltenheit der randomisierten Merkmale relevant. Beim „PriVaricator“ werden unter anderem die Abfrage von Plugins und die Erkennung von Schriften randomisiert. Da Entropie der Liste der Plugins auf mindestens 15,4 Bit und die Entropie der gemessenen Schriften auf mindestens 13,9 Bit geschätzt wurde [17], ist zu erwarten, dass im Idealfall mindestens 15,4 Bit Informationen verloren gehen. Werden allerdings die Abhängigkeiten zwischen A und B miteinbezogen, kann sich die Menge der verlorengehenden Information verringern. Da „PriVaricator“ als Forschungsimplementierung nicht versucht detaillierten Prüfungen und andere Wege der Plugin- oder Schriftbestimmung zu verhindern, ist anzunehmen, dass $P(B_x|A) \approx 1$ und somit kaum Informationen verloren gehen.

Das Ignorieren von Merkmalen betrifft aber alle Fingerprints, deren Merkmale andeuten, dass ein Randomisierer eingesetzt wird, egal ob diese Browser einen Randomisierer nutzen oder nicht. In einer allgemeineren Betrachtung ist also interessant, wie sich das Ignorieren von Merkmalen auf die Entropie der sich nun ergebenden Zufallsverteilung auswirkt. Würden bei $x\%$ der Nutzer der Fingerprint f_m statt f_n gemessen und dadurch durchschnittlich y Bit Information verlorengegangen, würde die Entropie der gemessenen Werte

$$\begin{aligned}
 & (100 - x)\% * E(I(f_n)) + x\% * E(I(f_m)) = \\
 & (100 - x)\% * E(I(f_n)) + x\% * (E(I(f_n)) - y) = \\
 & 100\% * E(I(f_n)) - x\% * y = \\
 & E(I(f_n)) - x\% * y
 \end{aligned}$$

betragen und somit um $x\% * y$ Bit fallen. Zu beachten ist hierbei, dass der Verlust der Information für jeden Nutzer nicht durch $-\log_2(P(B_x))$ sondern durch $-\log_2(P(B_x|A))$ bestimmt wird und deshalb ohne Kenntniss der Abhängigkeiten nicht bestimmt werden kann.

Exestiert allerdings ein B_x mit $P(B_x|A) \approx 1$ können Browsermerkmale ignoriert werden, ohne das preisgegebene Information relevant reduziert wird, da zu erwarten ist, dass $-\sum_{n=1}^N P(B_n|A)\log_2(P(B_n|A))$ Bit Informationen verloren gehen. Treten allerdings viele Ereignisse aus B häufig auf reduziert sich die preisgegebene Information für die randomisierten Nutzer und die Entropie der gemessenen Verteilung. Wenn zum Beispiel von 30% der Nutzer die Pluginauflistung ignoriert werden und dadurch durchschnittlich 10 Bit weniger an Informationen preisgegeben, würde die Entropie der Zufallsverteilung um $30\% * 10 \text{ Bit} = 3 \text{ Bit}$ sinken.

Zur Verfolgung von Nutzern ist diese Methode also zwar prinzipiell geeignet, bedeutet aber die Verschlechterung der allgemeinen Messergebnisse, wenn die ignorierten Merkmale nicht ersetzt werden. Daher kann davon ausgegangen werde, dass solche Maßnahmen nicht genutzt werden, wenn nur ein kleiner Teil der Nutzer Randomisierer einsetzt.

5.2.4 Zusammenfassung

Das Maß der Randomisierung ist ein zu vernachlässigendes Problem, da sehr kleine Mengen möglicher vom Randomisierer generierten Ergebnissen ausreichen, um zu verhindern, dass ein Fingerprint zwei mal ausgegeben wird.

Einen Fingerprintingalgorithmus, der nicht auf die Nutzung von Randomisierern eingestellt ist, kann von Randomisierern getäuscht werden. Kann dieser einen Randomisierer erkennen und auf diesen eingehen, ist es möglich, den Nutzer trotzdem zu verfolgen, wenn $\{A, B^*, M\}$ genug Information trägt.

Das Ignorieren von Merkmalen funktioniert zwar prinzipiell, um Nutzer trotz Randomisierung ihrer Browsermerkmale zu identifizieren, aber die Gesamtqualität der Nutzerverfolgung würde darunter leiden. Betrifft dies viele Nutzer und aussagekräftige Merkmale, könnte das Browserfingerprinting relevant behindert werden.

Der Ansatz der Randomisierung ist also vielversprechend und funktioniert für von der Industrie genutzten Fingerprintingskripte. Das Beobachten, Ignorieren und Fingerprinten von Änderungen im Fingerprint ist allerdings eine Schwäche dieser Methode, weswegen ein besonderer Augenmerk darauf gelegt werden sollte, dass die vorgenommenen Änderungen nicht beobachtbar oder fingerprintbar sind.

5.3 Skripte deaktivieren

Das Deaktivieren von Skripten in Browsern verhindert das skriptbasierte Browserfingerprinting komplett. Andere Formen des Browserfingerprintings sind allerdings dadurch nicht betroffen.

Die Browserinstallation gibt also weiterhin die Merkmale M_{passiv} , die mit passivem Browserfingerprinting erhoben werden können, und die Merkmale $M_{aktiv-skript}$, die aktiv, aber ohne Nutzung von Skripten erhoben werden können, preis. Zusätzlich wird noch das Merkmal $M_{noscript}$, das die Deaktivierung von Javascript anzeigt, bekannt und es könnten Merkmale M_{plugin} gefunden werden.

Es stehen also mindestens

EXPERIMENT

6.1 Simulation

ANMERKUNG: ohne Rechtschreibprüfung

ANMERKUNG: Es liegt noch keinerlei Implementierung vor.

ANMERKUNG: Wenn die Performance gut ist würde ich das gerne noch etwas erweitern

ANMERKUNG: Feedback zur Nutzbarkeit des Simulationssetups wäre sehr hilfreich.

Um die Auswirkungen von Gegenmaßnahmen gegen Browserfingerprints untersuchen zu können, ohne die Gegenmaßnahmen direkt zu implementieren, soll eine nicht-deterministische diskrete Simulation genutzt werden.

In Abschnitt 6.1.1 wird der grundsätzliche Aufbau der Simulation beschrieben. Um eine realitätsnahe Simulation zu erreichen, werden anschließend in Abschnitt 6.1.2 die Simulationsparameter angepasst. Abschließend werden in Abschnitt 6.1.3 die Ergebnisse eines Durchlaufes der Simulation ohne Gegenmaßnahmen dargestellt.

6.1.1 Aufbau der Simulation

Die Simulation soll auf Aktivitäten des Nutzers basieren und von diesen ausgehend die Trackingdatenbanken der Analyseserver aufbauen. Dazu werden in der Basissimulation nur naive Nutzer und naive Fingerprintingalgorithmen genutzt werden. Für spätere Nutzung werden spezialisierte Nutzer und Fingerprintingalgorithmen genutzt, die Grundfunktion der Simulation aber nicht verändert oder Änderungen explizit beschrieben.

Zeitscheiben

Zu Beginn jeder Zeitscheibe wird jedes User-Objekt mittels eines „Ticks“ benachrichtigt und vollzieht eine Aktion oder eine Reihe von Aktionen. Die User-Objekte werden dabei nach Reihenfolge ihres Indexes

abgearbeitet und jeweils auf die Beendigung ihrer Aktionen gewartet, bis der nächste Nutzer über den „Tick“ informiert wird. Um Verzerrungen aufgrund dieser Ausführungsreihenfolge zu vermeiden, sollten pro „Tick“ nur wenige Aktionen von einem User-Objekt ausgeführt werden.

Nach Beendigung jeder Zeitscheibe werden die Statistiken für diese Zeitscheibe angefertigt und die nächste Zeitscheibe angestoßen. Ist eine feste Menge an Zeitscheiben durchlaufen, wird die Simulation beendet und die Nachverarbeitung der Ergebnisse begonnen.

Nutzer

Der Nutzer soll durch ein User-Objekt repräsentiert werden, das folgende Aufgaben hat:

- Den Zustand der Merkmale speichern.
Die Merkmale repräsentieren die Merkmale von Browsern im abstrakten. Diese sind unbenannt und es soll kein direkter Zusammenhang zu konkreten Merkmalen von Browsern hergestellt werden. Alle Merkmale sollen jeweils 2 Byte umfassen und dabei die Werte 0 bis $(2^{16} - 3) = 65533$ annehmen können, um Raum für die Spezialwerte „NULL“ und „wechselnd“ reservieren zu können. Kann ein Merkmal von keinem Fingerprinting-Objekt gelesen werden, ist es unnötig, dieses zu speichern. Um die Menge des belegten Arbeitsspeichers zu reduzieren wird dies differenziell zu Referenzmerkmalsets gespeichert.
- Eine Browsing-session starten, fortführen oder beenden.
Das User-Objekt speichert in seinem Zustand, ob eine Browsing-session im Gange ist. Zusätzlich speichert es die Wahrscheinlichkeit, mit der eine Browsing-session begonnen beziehungsweise beendet wird. Ist bei Benachrichtigung eines „Ticks“ keine Browsing-session im Gange, wird eine Browsing-session begonnen oder keine Aktion getätigt. Ist eine Browsing-session im Gange, wird diese entweder beendet oder fortgesetzt.
- Browsingprofil speichern.
Das Browsingprofil gibt die Wahrscheinlichkeiten an mit denen eine neue Domain besucht oder dieselbe Domain erneut besucht wird. Zusätzlich wird die Wahrscheinlichkeit für das Starten und Beenden einer Browsing-session gespeichert. Um die Menge des belegten Arbeitsspeichers zu reduzieren wird dies differenziell zu Referenzbrowsingprofilen gespeichert.
- Die Nutzer-ID speichern.
Die Nutzer-ID identifiziert den Nutzer eindeutig. Dieses Merkmal wird bei einem Webseitenbesuch mitgesendet, aber nicht in einem Fingerprint verwertet.

Browsing-sessions

Browsing-sessions sind abstrakt und die notwendigen Zustände werden im User-Objekt gespeichert. Wird eine Browsing-session gestartet, wird eine Domain besucht und anschließend die Kontrolle zurückgegeben. Wird eine Browsing-session fortgesetzt wird die zuletzt besuchte Domain erneut besucht oder eine neue Domain besucht. Welche Aktion getätigt wird mittels des Browsingprofils des User-Objekts zufällig ausgewählt.

Domains

Die Domain-Objekte repräsentieren die Webseiten im Internet. Diese referenzieren lediglich eine Menge von Fingerprintern und leiten den Besuch an diese weiter. Um Speicherplatz zu sparen, werden nur Domains betrachtet, die mindestens einen Fingerprint referenzieren.

Initialisierung der User-Objekte

Vor Beginn der Simulation müssen die User-Objekte initialisiert werden. Dazu müssen die Merkmale und das Browsingverhalten generiert werden.

Dazu werden zunächst die Referenzmerkmalsets und Referenzbrowsingprofile als Simulationsparameter geladen. Diese enthalten für jeden Eintrag ein Standardwert, eine Wahrscheinlichkeit für abweichende Werte und eine Gewichtung. Anschließend wird von den gewichteten Referenzmerkmalsets und Referenzbrowsingprofilen ein zufälliges ausgewählt und abweichende Einträge ausgewürfelt. Die abweichenden Werte, das Referenzmerkmalset und das Referenzbrowsingprofil wird nun im User-Objekt gespeichert.

Fingerprinter

Ein Fingerprinter fragt bei dem Besuch eines User-Objektes eine Menge von Merkmalen des User-Objektes und die Nutzer-ID des Nutzers ab. Diese Daten und die Referenzierende Seite werden abgespeichert und werden nach der Simulation analysiert.

6.1.2 Bestimmung der Simulationsparameter

6.1.3 Ergebniss ohne Gegenmaßnahmen

ZUSAMMENFASSUNG

Literaturverzeichnis

- [1] URL <https://browserhacks.com/Zuletztbesuchtam>.
- [2] URL <http://www.ghostery.comZuletztbesuchtam>.
- [3] URL <https://www.lalit.org/lab/javascript-css-font-detect/>.
- [4] URL <http://www.trusteer.com/blog/how-fraudsters-are-disguising-pcs-fool-device-fingerprints>.
- [5] URL <https://http://www.w3.org/community/pua/wiki/Draft>.
- [6] Erwan Abgrall, Yves Le Traon, Martin Monperrus, Sylvain Gombault, Mario Heiderich, and Alain Ribault. Xss-fp: Browser fingerprinting using html parser quirks. *arXiv preprint arXiv:1211.4812*, 2012.
- [7] Gunes Acar. Obfuscation for and against device fingerprinting position paper for symposium on obfuscation new york university, february 15, 2014.
- [8] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. The web never forgets: Persistent tracking mechanisms in the wild.
- [9] Gunes Acar, Marc Juarez, Nick Nikiforakis, Claudia Diaz, Seda Gürses, Frank Piessens, and Bart Preneel. Fpdetective: Dusting the web for fingerprinters. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 1129–1140. ACM, 2013.
- [10] Ahmed Awad E Ahmed and Issa Traore. Detecting computer intrusions using behavioral biometrics. In *PST*, 2005.
- [11] Francesco Bergadano, Daniele Gunetti, and Claudia Picardi. User authentication through keystroke dynamics. *ACM Transactions on Information and System Security (TISSEC)*, 5(4):367–397, 2002.
- [12] Frédéric Besson, Nataliia Bielova, and Thomas Jensen. Enforcing Browser Anonymity with Quantitative Information Flow. Technical Report RR-8532, INRIA, 2014. URL <http://hal.inria.fr/hal-00984654>.
- [13] Daniel Bilar et al. Statistical structures: Fingerprinting malware for classification and analysis. *Proceedings of Black Hat Federal 2006*, 2006.
- [14] Sergey Bratus, Cory Cornelius, David Kotz, and Daniel Peebles. Active behavioral fingerprinting of wireless devices. In *Proceedings of the first ACM conference on Wireless network security*, pages 56–61. ACM, 2008.
- [15] Ralph Broenink. Using browser properties for fingerprinting purposes. In *16th biannual Twente Student Conference on IT*, pages 169–176, 2012.
- [16] Timothy G Buchman, Bernard Roizman, Garrett Adams, and Beth Hewitt Stover. Restriction endonuclease fingerprinting of herpes simplex virus dna: a novel epidemiological tool applied to a nosocomial outbreak. *Journal of Infectious Diseases*, 138(4):488–498, 1978.
- [17] Peter Eckersley. How unique is your web browser? In *Privacy Enhancing Technologies*, pages 1–18. Springer, 2010.
- [18] Mark Fioravanti. Client fingerprinting via analysis of browser scripting environment. *SANS Information Security Reading Room*, 2010.
- [19] Daniel Genkin, Adi Shamir, and Eran Tromer. Rsa key extraction via low-bandwidth acoustic cryptanalysis. *IACR Cryptology ePrint Archive*, 2013:857, 2013.

- [20] Miroslav Goljan, Jessica Fridrich, and Tomáš Filler. Large scale test of sensor fingerprint camera identification. In *IS&T/SPIE Electronic Imaging*, pages 72540I–72540I. International Society for Optics and Photonics, 2009.
- [21] Katerina Goseva-Popstojanova, Brandon Miller, Risto Pantev, and Ana Dimitrijević. Empirical analysis of attackers activity on multi-tier web systems. In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pages 781–788. IEEE, 2010.
- [22] Kim Griggs, Laurie M Bridges, and Hannah Gascho Rempel. Library/mobile: tips on designing and developing mobile web sites. *Code4lib journal*, 8, 2009.
- [23] Michael Risher Hong Kaing and Brian Schulte. User tracking: Persistent cookies and browser fingerprinting. 2013. URL http://cs.gmu.edu/~yhwang1/INFS612/2013_Spring/Projects/Final/2013_Spring_PGN_5_final_report.pdf.
- [24] Peter H Horadan, Matthew R Shanahan, and Mark B Upson. Method and apparatus for correlating multiple cookies as having originated from the same device using device fingerprinting, May 24 2011. US Patent App. 13/114,780.
- [25] Anil K Jain and Sharath Pankanti. Beyond fingerprinting. *Scientific American*, 299(3):78–81, 2008.
- [26] Samy Kamkar. Evercookie. URL: <http://samy.pl/evercookie>, 2010.
- [27] Dustin Lee, Jeff Rowe, Calvin Ko, and Karl Levitt. Detecting and defending against web-server fingerprinting. In *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*, pages 321–330. IEEE, 2002.
- [28] Clemens Kolbitsch Benjamin Livshits, Benjamin Zorn, and Christian Seifert. Rozzle: De-cloaking internet malware. 2011.
- [29] Jonathan R Mayer. Any person... a pamphleteer”: Internet anonymity in the age of web 2.0. *Undergraduate Senior Thesis, Princeton University*, 2009.
- [30] Jonathan R Mayer and John C Mitchell. Third-party web tracking: Policy and technology. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 413–427. IEEE, 2012.
- [31] Katherine McKinley. Cleaning up after cookies. Technical report, Technical report, iSEC PARTNERS. <https://www.isecpartners.com/research/white-papers/cleaning-up-after-cookies.aspx> zuletzt besucht am, 2010.
- [32] Keaton Mowery and Hovav Shacham. Pixel perfect: Fingerprinting canvas in html5. *Proceedings of W2SP*, 2012.
- [33] Keaton Mowery, Dillon Bogenreif, Scott Yilek, and Hovav Shacham. Fingerprinting information in javascript implementations. *Proceedings of W2SP*, 2011.
- [34] Martin Mulazzani, Philipp Reschl, Markus Huber, Manuel Leithner, Sebastian Schrittwieser, Edgar Weippl, and FH Campus Wien. Fast and reliable browser identification with javascript engine fingerprinting. In *Web 2.0 Workshop on Security and Privacy (W2SP)*, volume 5, 2013.
- [35] D Neuhaus, H Kühn, J-G Kohl, P Dörfel, and T Börner. Investigation on the genetic diversity of *phragmites* stands using genomic fingerprinting. *Aquatic Botany*, 45(4):357–364, 1993.
- [36] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 541–555. IEEE, 2013.
- [37] Nick Nikiforakis, Wouter Joosen, and Benjamin Livshits. Privaricator: Deceiving fingerprinters with little white lies. 2014.
- [38] Athanasios N Papanicolaou, Jimmy F Fox, and John Marshall. Soil fingerprinting in the palouse basin, usa, using stable carbon and nitrogen isotopes. *International Journal of Sediment Research*, 18(2):278–284, 2003.
- [39] MB Pool. Meantime: non-consensual http user tracking using caches (2000). *Blog entry. Referenced*, 2006.

- [40] Matthew Smart, G Robert Malan, and Farnam Jahanian. Defeating tcp/ip stack fingerprinting. In *Usenix Security Symposium*, 2000.
- [41] Lance Spitzner. Passive fingerprinting. *FOCUS on Intrusion Detection: Passive Fingerprinting (May 3, 2000)*, pages 1–4, 2000.
- [42] Henning Tillmann. Browserfingerprinting: Tracking ohne spuren zu hinterlassen. 2013.
- [43] Thomas Unger, Martin Mulazzani, Dominik Fruhwirt, Markus Huber, Sebastian Schrittwieser, and Edgar Weippl. Shpf: Enhancing http (s) session security with browser fingerprinting. In *Availability, Reliability and Security (ARES), 2013 Eighth International Conference on*, pages 255–261. IEEE, 2013.
- [44] Mick Vaites. The effectiveness of a browser fingerprint as a tool for tracking. 2013.
- [45] Gérard Wagener, Alexandre Dulaunoy, and Radu State. Torinj: Automated exploitation malware targeting tor users. *arXiv preprint arXiv:1208.2877*, 2012.
- [46] Zachary Weinberg, Eric Yawei Chen, Pavithra Ramesh Jayaraman, and Collin Jackson. I still know what you visited last summer: Leaking browsing history via user interaction and side channel attacks. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 147–161. IEEE, 2011.
- [47] Zhihong Xu. Fingerprinting global climate change and forest management within rhizosphere carbon and nutrient cycling processes. *Environmental Science and Pollution Research*, 13(5):293–298, 2006.
- [48] Ting-Fang Yen, Yinglian Xie, Fang Yu, Roger Peng Yu, and Martin Abadi. Host fingerprinting and tracking on the web: Privacy and security implications. In *NDSS*, 2012.

