



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



[BACHELOR/MASTER] THESIS

[YOUR TITLE]

[YOUR NAME]

Erlangen, July 25, 2014

Examiner: [YOUR EXAMINER (probably Prof. Dr. Felix Freiling)]
Advisor: [YOUR ADVISOR]

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, July 25, 2014

[YOUR NAME]

Zusammenfassung

Zusammenfassung auf Deutsch Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Abstract

Zusammenfassung auf Englisch Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

CONTENTS

1	Einleitung	1
2	Browserfingerprinting	3
2.1	Modell	4
2.2	Nutzungsmöglichkeiten	5
2.2.1	Ersatz für Cookies	5
2.2.2	Zusätzliches Absichern von Sessions	5
2.2.3	Accounts an Nutzer binden	6
2.2.4	Polizei und Geheimdienstarbeit	6
2.3	Passives und aktives Fingerprinting	6
2.3.1	Passives Fingerprinting	7
2.3.2	Aktives fingerprinting	7
2.4	Gegenmaßnahmen	8
2.4.1	Selbstbeschreibungen einschränken	8
2.4.2	Standardisierung von Systemen	9
2.4.3	Seitenkanäle abdichten	9
2.4.4	Skriptsprachen einschränken	9
2.4.5	Traffic normalisieren	10
2.4.6	Fingerprints ändern	10
2.4.7	Entzug der Kommunikation	10
2.4.8	Fälschung von Fingerprints	11
2.5	Zusammenfassung	11
3	Gegenmaßnahmen	13
4	theoretische Betrachtung	15
5	Simulation	17
6	Zusammenfassung	19

EINLEITUNG

BROWSERFINGERPRINTING

Das Fingerprinten von Programmen ist eine schon länger bekannte Technik, die genutzt wird, um gezielt Angriffe auf Computersysteme durchzuführen. Dabei werden Eigenheiten der Programme 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. Dabei werden möglichst unveränderliche und nicht unterdrückbare Eigenheiten der Programme genutzt und miteinander kombiniert, um diese Informationen zu gewinnen. Diese Eigenschaften können wie der namensgebenden Fingerabdruck schwer fälschbar sein und auch unfreiwillig abgegeben werden.

Das Fingerprinten eines Browsers funktioniert im Bereich Datensammlung ähnlich wie das Fingerprinten eines SSH Servers oder Webservers. Ein Browser ist allerdings ein komplexeres System als ein SSH Client, da der Browser wesentlich mehr Eigenheiten, Einstellungs-, Erweiterungs- und Interaktionsmöglichkeiten hat. Deshalb lassen sich beim Fingerprinten eines Browsers besonders viele Informationen gewinnen. Auch wenn diese Informationen uninteressant sind, erlauben sie Browserinstallationen zu unterscheiden. Stehen genug Informationen zur Verfügung, um eine Browserinstallation von allen anderen Browserinstallationen zu unterscheiden, kann diese sogar identifiziert werden.

Die Möglichkeit, Browserfingerprinting zur Identifizierung von Browserinstallationen zu nutzen, wurde 2010 in einem Paper von Eckersley untersucht. Eckersley hat ein Modell des Browserfingerprintings formuliert 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 viele Browserinstallationen mittels Browserfingerprinting identifizierbar sind.

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 Neukonfigurationsänderungen ändert. Um trotz dieser Änderungen Browserinstallationen identifizieren zu können, wurden Algorithmen eingesetzt, die kleine Änderungen des Fingerprints entdecken und mit dem originalen Fingerprint in

Zusammenhang stellen können.

Die tatsächliche Nutzung von Browserfingerprinting wurde in der Studie "Cookies Monster: Exploring the Ecosystem of Web-based Device Fingerprinting" untersucht. 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 Top 10000 Alexa-Seiten Fingerprintingskripte gefunden. Diese Zahl ist aber eine untere Grenze, da mit dem eingesetzten Crawler nur bestimmte Browserfingerprintingskripte erkannt werden konnten und nicht alle Unterseiten überprüft wurden. Zusätzlich können Trackingseiten auch auf anderen Seiten eingebunden werden und so auch Besuche anderer Domains betreffen. Die Frage wofür die gefundenen Browserfingerprints genutzt werden konnte allerdings nicht abschließend geklärt werden.

Um einen Überblick über den Forschungsstand zum Thema Browserfingerprinting zu geben, wird zunächst in Abschnitt 2.1 ein mathematisches Modell und in Abschnitt 2.2 die Nutzung des Browserfingerprintings vorgestellt. Anschließend wird in Abschnitt 2.3 das aktive und passive Fingerprinting vorgestellt und auf Merkmale eingegangen, die für das Browserfingerprinting genutzt werden. Abschließend werden in Abschnitt 2.4 bekannte Maßnahmen gegen Browserfingerprinting dargestellt.

2.1 Modell

In der Studie "panopticlick" von Eckersley wurde ein mathematisches Modell des Fingerprintingprozesses genutzt, um Aussagen über die Möglichkeit des Browserfingerprintings zu treffen, auf dem diese Arbeit ebenfalls basiert.

Bei diesem Modell wird die als Fingerprint gesammelte Information als Ergebnis eines Zufallsexperiments interpretiert. Das Zufallsexperiment ist dabei die Installation, Konfiguration und Nutzung des Browsers. Der Inhalt der gesammelten Informationen spielt dabei keine Rolle, da Aussagen über das Zufallsexperiment getroffen werden sollen.

Die Messung eines Fingerprints mit einem Fingerprintingalgorithmus entspricht $F(\cdot)$ und der gemessene Fingerprint für eine Browserinstallation x entspricht $F(x)$. Die Verteilung der Fingerprints $f_n, n \in [0, 1, \dots, N]$ hat dabei die Zufallsdichte $P(f_n)$. Die Zufallsverteilung und sogar die Anzahl der Fingerprints N ist dabei eine Unbekannte. Die von einem Fingerprint f_n preisgegebene Information I , von Eckersley surprisal genannt, ist durch die Formel $I(F(x) = f_n) = -\log_2(P(f_n))$ gegeben. Die Entropie der Zufallsverteilung entspricht dem Erwartungswert der preisgegebenen 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 diese als Teilmessungen mit verschiedenen Algorithmen $F_s(\cdot)$ behandeln zu können. Da die einzelnen Messungen voneinander abhängig sein können, wie zum Beispiel der Useragent und der Browsertyp, können diese nicht einfach addiert werden. Sind die Abhängigkeiten zwischen 2 Messungen $F_s(\cdot)$ und $F_t(\cdot)$ bekannt, kann eine Berechnung der insgesamt preisgegebenen 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 angenommen werden.

Dieses Modell wurde von Eckersley genutzt, um mittels einer Messung vieler Browserfingerprints und eines konkreten Fingerprintingalgorithmus $F(\cdot)$ auf Eigenschaften der Zufallsverteilung $P(f_n)$ 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. Eckersley weist aber schon bei dem Erstellen des Modells darauf hin, dass eine simple Berechnung von $2^{33} = 8.589.934.592 > 7,2 \text{ Milliarden} \approx \text{Anzahl Menschen}$ auf diesem Planeten naiv ist. Zudem kann 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 nach naiver Rechnung bereits 10 Bit Entropie aus, um die Browser zu identifizieren, da $2^{10} = 1024 > 1000$.

Eine weitere Betrachtungsweise, die aus der Perspektive eines einzelnen Users hilfreich ist, ist das Anonymity-set. Ein Anonymity-set ist die Menge der Browserinstallationen, die einen identischen Fingerprint beziehungsweise identische Merkmale haben. Teilt ein Browser ein Anonymity-set mit einem anderen Browser, kann er also von diesem nicht unterschieden, also auch nicht eindeutig identifiziert werden. Wird trotzdem eine Identifizierung versucht, gelingt diese nach einfacher Rechnung mit der Wahrscheinlichkeit $1/|\text{Anonymityset}|$.

2.2 Nutzungsmöglichkeiten

In der Studie "Cookieles Monster: Exploring the Ecosystem of Web-based Device Fingerprinting" wurde der Umfang des Browserfingerprintings erforscht. Über die Ausschöpfung der in der Studie gefundenen Nutzungsmöglichkeiten konnte nur in Einzelfällen eine Aussage getroffen werden. Auch bei diesen Aussagen musste sich auf Eigenaussagen und Vermutungen verlassen werden, da die tatsächliche Nutzung von Browserfingerprinting nicht von außen überprüft werden kann.

Bekannte Nutzungsmöglichkeiten dieser Form der Nutzerwiederidentifizierung ist die Nutzung als Ersatz für Cookies, das zusätzliche Absichern von Sessions, die Bindung von Nutzern an Services, das Identifizieren von bössartigen Nutzern und in der Polizei oder Geheimdienstarbeit. Diese Nutzungsmöglichkeiten und die prinzipielle Funktionsweise werden in den nächsten Abschnitten vorgestellt.

2.2.1 Ersatz für Cookies

Es gibt zwar verschiedene Arten, Daten in einem Browser abzulegen, die Cookies direkt ersetzen können und sogar über "Evercookie" gesammelt ansprechbar sind, aber Browserfingerprinting wird trotzdem genutzt um Cookies zu ersetzen. Dabei wird aber das Ablegen von Daten in dem Browser vermieden und der Nutzer kann die Cookies nicht einfach löschen. Aus diesem Grund sind Privatmodi von Browsern für Browserfingerprinting anfällig, wenn diese lediglich verhindern, dass Daten im Browser abgelegt werden. Zusätzlich ist die Wiederidentifizierung über Browserfingerprints unauffälliger als über Cookies, da dies nicht zwangsweise Spuren im Browser hinterlässt. Browserfingerprinting kann auch in Kombination mit Cookies genutzt werden, um Nutzer trotz des Löschens aller Cookies wiederzuerkennen und die Cookies wiederherzustellen.

Damit die Nutzung von Browserfingerprints in diesem Kontext Sinn macht, muss davon ausgegangen werden, dass der Browser des Nutzers keine Cookies erlaubt oder Cookies gelöscht werden. Der Nutzer somit seinen Willen ausdrückt, nicht wiedererkannt zu werden und eine Nutzung zu diesem Zweck kann illegal sein. Eckersley gibt an, 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 Trackingunternehmen zugeschrieben wird unter Druck zu stehen, Nutzer auch gegen ihren Willen zu verfolgen, denn Cookies werden zunehmend als Problem behandelt und vermieden.

2.2.2 Zusätzliches Absichern von Sessions

Sollen eine Reihe von Aktionen ausgeführt, die durch ein Login geschützt sind, wird oft eine Session genutzt, um mit einem Login auszukommen. Bei einer Session wird zu Beginn der Session ein Geheimnis generiert, das 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 ein Angreifer dieses Geheimnis und nutzt dieses um sich gegenüber dem Server auszuweisen und auf die mit der Session verknüpften Daten zuzugreifen. Um Nutzer gegen einen solchen Angriff zu schützen, kann der Server zusätzlich den Fingerprint des Clients speichern und überprüfen. Ändert sich der Fingerprint auffällig stark, kann angenommen werden, dass eine unzulässige Übertragung der Session stattgefunden hat, und die Session geschlossen werden. PHP-Sessions können auf diese Weise erweitert und gegen Sessionhijacking geschützt werden. Da das Prinzip kann aber auch auf

Sessions angewendet werden, bei denen nur wenige Informationen zu Verfügung stehen, wie SSH oder TCP/IP Sessions.

Die Sicherheit eines solchen Systems beruht auf den Annahmen, dass der Angreifer weder den Fingerprint reproduzieren kann, da der Fingerprint des Opfers unbekannt oder nicht reproduzierbar ist, noch durch Zufall denselben Fingerprint des Opfers hat. Da nicht erwiesen ist, dass diese Annahmen zutreffen, kann Sessionhijacking nicht ausgeschlossen, aber immerhin die Komplexität eines solchen Angriffs erhöht werden. Ändert sich der Fingerprint des legitimen Nutzers zu stark und die Session wird unberechtigt geschlossen, ist der Nutzer gezwungen, sich erneut einzuloggen.

2.2.3 Accounts an Nutzer binden

Soll die Nutzung eines Accounts auf einen oder wenige Nutzer beschränkt werden, kann versucht werden dies über deren Browserfingerprint durchzusetzen. Dazu speichert der Server einen oder mehrere Fingerprints, die mit den Accounts verknüpft sind. Um Fingerprints mit einem Account zu verknüpfen, kann zum Beispiel der Fingerprint des ersten Logins gespeichert oder zusätzlichen Authentifikationsmethoden genutzt werden. Sind die Fingerprints mit dem Account verknüpft, kann nun bei jedem weiteren Login-Versuch überprüft werden, ob der Fingerprint des Clients mit den verknüpften Fingerprints übereinstimmt. Ist dies nicht der Fall, kann der Login verweigert, zusätzliche Authentifikationsmethoden gefordert oder ein Einbrucherkennungssystem benachrichtigt werden.

Praktisch kann dies im Hochsicherheitsbereich genutzt werden, bei dem angenommen werden kann, dass nur ein Nutzer einen Account nutzen soll und zusätzliche Authentifikationsmethoden existieren, falls sich der Fingerprint des legitimen Nutzers ändert. Ein Beispiel für einen solchen Hochsicherheitsbereich ist das Onlinebanking, bei dem auch zusätzliche Authentifikationsmethoden wie TANs existieren. Aber auch in Bereichen mit niedrigen Sicherheitsanforderungen, wie dem bezahlten Mediastreaming, kann eine Bindung von Accounts an Nutzer angewendet werden. Bei diesem haben die Nutzer ein Interesse an der Mehrfachnutzung eines Accounts. Die Anbieter hingegen haben ein Interesse daran, dass möglichst viele Accounts erstellt werden. Um zu verhindern, dass sich eine große Menge von Nutzern unter einem Account einloggt und gleichzeitig ein Nutzer sich 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 zum Beispiel der Login verweigert wird.

2.2.4 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.

Ob Polizei oder Geheimdienste Browserfingerprinting nutzen, ist letztlich nicht relevant, da angenommen wird, dass diese das tun. Dementsprechend ist das Anonymisierungswerkzeug TOR auch auf Deanonymisierungstechniken vorbereitet, auch wenn sie Browserfingerprinting verwenden. Die für die Polizeiarbeit relevante Frage der Beweiskraft von Browserfingerprints ist dabei noch ungeklärt.

2.3 Passives und aktives Fingerprinting

Als Unterscheidung zwischen aktivem und passivem Browserfingerprinting findet sich die Definition, dass aktives Fingerprinting clientseitige Skriptsprachen wie Javascript nutzt, passives hingegen nicht. 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. Dazu werden bereits Unterschiede in Syntaxparsing der Browser genutzt, um ohne Javascript zwischen Browser differenzieren zu können. In dieser Arbeit soll dieser Fall allerdings unter aktives Browserfingerprinting fallen, wobei eine Definition für aktives und passives Browserfingerprinting genutzt wird, wie sie auch in andern Fingerprintingkontexten verwendet wird. In dieser Definition ist Fingerprinting passiv, wenn die Kommunikation zwischen Client und Server nicht verändert wird, und ansonsten aktiv.

2.3.1 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:

... ... BLOCK WITH PASSIVE FINGERPRINTING ATTRIBUTES ...

Diese Merkmale können auch beim aktiven Fingerprinting ausgelesen werden, deswegen fallen beim passiven Fingerprinting im Vergleich zum aktiven Fingerprinting weniger Informationen an.

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 "Cookieles 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 fingerprinten 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.

2.3.2 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 Protokollen gefragt wird. Eine weitergehende Möglichkeit ist Eigenheiten auszuforschen, indem Maximalgrößen getestet werden oder Syntax verwendet wird, die undefiniertes oder nicht einheitliches Verhalten aufweist. Unterstützt das zu fingerprintende System die Ausführung von Code, kann dieser auch eventuell dazu genutzt werden, das System zu fingerprinten.

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 und das Benchmarken von Javascriptfunktionen, um auf den genutzten Javascriptengine zu schließen. Für das Zurücksenden der gesammelten Daten kann auf dafür vorgesehen Funktionen wie AJAX zurückgegriffen werden, aber auch andere verstecktere Kanäle könnten genutzt werden.

Einer der bekanntesten von Browsern unterstützten Skriptsprachen ist Javascript. Diese allein war im Jahr XXXX von XX % der Nutzer aktiviert und es gibt weitere Sprachen wie Flash, Silverlight oder Java, die zum Fingerprinten geeignet sind. Beim Browserfingerprinting kann also davon ausgegangen werden, dass es mittels solcher Skripte möglich ist, ohne allzu viele Nutzer auszulassen. Um sicherzustellen, dass auch

tatsächlich alle Nutzer Skripte aktiviert haben, kann 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, sind unter anderem:

... .. BLOCK WITH ACTIVE NON-SCRIPTING FINGERPRINTING ATTRIBUTES ...

... .. BLOCK WITH SCRIPTING FINGERPRINTING ATTRIBUTES ...

Das Browserfingerprinting mittels Skripten steht, wohl aufgrund der Menge an erhebbaren Informationen und der breiten Verfügbarkeit von Skripten, im Fokus von Industrie und Forschung. Das Interesse der Industrie ist daran erkennbar, dass viele Browserfingerprintingbibliotheken ausschließlich in Javascript implementiert sind und somit ohne aktivierte Skripte noch nicht einmal lauffähig sind. Das Interesse der Forschung dagegen ist daran erkennbar, dass der Fingerprintingmechanismus, der in der Diplomarbeit von Tillmann zur Untersuchung von Browserfingerprints eingesetzt wurde, ebenfalls auf Javascript basiert. Ebenso konnte der Crawler, mit dem in der Studie "Cookieles Monster: Exploring the Ecosystem of Web-based Device Fingerprinting" nach Browserfingerprinting gesucht wurde, nur auf Skripten basierendes Browserfingerprinting erkennen.

2.4 Gegenmaßnahmen

Gegen Browserfingerprinting und das Fingerprinten von Programmen existieren mehrere Vorschläge zu 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.

2.4.1 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, dies auf diesen Informationen basieren.

Das führt allerdings durch das "Privacy paradox" nicht zwingend zum Erfolg, sondern kann auch den eigentlichen Zielen entgegenwirken. 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. Im Extremfall könnte ein Browser, der als einziger keinerlei Informationen preisgibt, eben genau darüber identifiziert werden. 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 fällt auf 0 Bit, wenn alle Browser diese Information geheim halten.

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

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

Ein weiterer Punkt, an dem viele Informationen über die Browserinstallation gesammelt werden können, ist die Abfrage von Daten über Skript-APIs. Wird Flash genutzt, kann sogar der CPU Typ des Computers, auf dem Browser ausgeführt wird, abgefragt werden. In dem TOR-Browser-Bundle wird eine Zwischenlösung zwischen der kompletten Geheimhaltung und der kompletten Preisgabe von Informationen angewendet. Bei dieser dürfen nur maximal 10 Sprachen über die dafür vorgesehene Javascript-API abgefragt werden. 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.

2.4.2 Standardisierung von Systemen

Da der Fingerprint eine Messung einer Browserinstallation darstellt, kann dieser identische Browserinstallationen nicht unterscheiden. Je Ähnlicher sich Browserinstallationen sind, desto weniger Browserinstallationen können unterschieden werden. 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 sind, desto stärker ist der Nutzer beeinträchtigt, da dieser keine freie Wahl zwischen Programmen oder Konfigurationsoptionen treffen kann.

Mit diesem Effekt erklärt XXX die im Vergleich zu anderen Browsern geringe Informationsmengen, die von manchen Browsern preisgegeben werden. Konkret nennt XXX 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.

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 hier ein Standard gesetzt, der von einer großen Gruppe von Personen genutzt wird. 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.

2.4.3 Seitenkanäle abdichten

Informationen über einen Browser können von diesem nicht nur absichtlich, sondern auch unbeabsichtigt über Seitenkanäle übermittelt werden. Dies ist gewöhnlicherweise nicht nur unbeabsichtigt, sondern auch ungewollt, 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, 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.

2.4.4 Skriptsprachen einschränken

Die große Menge von Handlungsmöglichkeiten, die ein Fingerprintingskript hat, wenn es im Browser ausgeführt wird, erlaubt nicht nur das Sammeln von Informationen aus Seitenkanälen und APIs, sondern bietet auch die Möglichkeit, diese Informationen an einen Server zu senden.

Ein Ansatz, dies zu verhindern, ist die Ausführung von Skripten zu verweigern. 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 ist die Mächtigkeit der Skriptsprachen zu reduzieren. 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 kann. 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 zu nutzen, die nur einfache Skripte nutzen und trotzdem die Menge an Informationen zu reduzieren, die über die Browserinstallation in Erfahrung gebracht werden können.

2.4.5 Traffic normalisieren

Eine Möglichkeit, die vorgeschlagen wurde, um TCP/IP Fingerprinting von Servern zu verhindern, ist das Normalisieren des Traffics. 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.

Sollte dieses Konzept auf Browserfingerprints übertragen werden, müsste der HTTP-Traffic abgefangen 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.

2.4.6 Fingerprints ändern

Soll ein Browser nicht wideridentifiziert 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 nichtmehr 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, weswegen auf eine detaillierte Beschreibung verzichtet werden muss.

2.4.7 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. Um die Analyse durch diese Dienstleister komplett zu unterbinden, wurde das Plugin Ghostery entwickelt, dass diese eingebundenen Elemente erkennt und entfernt. Dadurch wird nicht mit dem Trackingdienstleister kommuniziert und dieser kann keine Analyse durchführen.

2.4.8 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. Mit Analysen, die kaum veränderbare Eigenheiten im Syntaxparsing des Browsers nutzen, um Aussagen über den Browser zu tätigen, können Falschinformationen über zum Beispiel den Browsertyp erkannt werden. Aber auch diese Analysen können aber wiederum getäuscht werden, indem die Installation des Opfers komplett nachgebildet wird.

2.5 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 mit wenig Informationen.

Es gibt eine Reihe bekannter Ansätze, um Browserfingerprinting zu verhindern, diese sind bei Weitem nicht so gut erforscht wie das Browserfingerprinting selbst. Die Gegenmaßnahmen schränken den Nutzer teilweise stark ein oder sind nicht implementiert. Eine Gegenmaßnahme, die ein eindeutiger Favorit ist, ist nicht bekannt.

GEGENMASSNAHMEN

4

THEORETISCHE BETRACHTUNG

5

SIMULATION

ZUSAMMENFASSUNG
