Web Privacy

Thomas Mauerer

Betreuer: Marcel von Maltitz
Seminar Future Internet
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: thomas.mauerer@tum.de

KURZFASSUNG

Tracking beschreibt die Methode, Daten über einen Nutzer in Erfahrung zu bringen. [1] Das Ziel ist, Nutzer wieder identifizieren zu können und umfangreiche Profile zu erstellen. Man unterscheidet zwischen expliziten und schließenden Verfahren. [1] Bei expliziten Verfahren werden Informationen direkt auf dem Computer des Nutzers gespeichert, wohingegen schließende Verfahren ihre Informationen aus Systemeigenschaften und durch den Nutzer vorgenommene Konfigurationen erhalten. Durch Einbinden von dritten Parteien, kann auch jenen Tracking ermöglicht werden. In dieser Arbeit werden verschiedene Tracking-Verfahren vorgestellt und es wird aufgezeigt, wie sich der einzelne Nutzer dagegen schützen kann. Alle genannten Verfahren werden im Zuge von Webinteraktion eingesetzt, d.h. sie beschreiben Web-Tracking. Die Themen App-Tracking und Smart-TV-Tracking werden in dieser Arbeit nicht behandelt. Zur besseren Veranschaulichung ist eine Beispielimplementierung vorhanden, welche unterschiedliche Tracking-Verfahren kombiniert.

Schlüsselworte

Cookies, HTML5 Storage, Evercookies, Fingerprinting

1. EINLEITUNG

Der Begriff „Web Privacy" heißt übersetzt so viel wie „Datenschutz im Internet". Konkret geht es bei diesem Thema also darum, welche Informationen über einen Nutzer gesammelt werden und wem diese Informationen zufließen. [20] Diese Seminararbeit beschäftigt sich mit verschiedenen Tracking-Verfahren und möglichen Schutzmaßnahmen dagegen.

Web-Tracking hat neben der eindeutigen Wiedererkennung eines Nutzers auch das Ziel, umfangreiche Profile über diese zu erstellen. Zu einem Profil gehören zum Beispiel Alter, Beruf, Geschlecht, Hobbies, Interessen, politische Einstellung, sexuelle Orientierung, Trinkgewohnheiten, etc. [1] Die gesammelten Daten dienen heutzutage als eine Art Rohstoff, d.h. die Daten können zu Geld verwertet werden. Beispielsweise können dadurch neue Kunden gewonnen werden. Auch ist es für Unternehmen möglich, sich am Markt zu etablieren, da dies ein ständiges Feedback der Nutzer voraussetzt, welches durch Tracking gewonnen wird. Eine weitere Einnahmequelle ist zielgerichtete Werbung. Damit dieses Konzept funktioniert, muss man zuerst die Interessen der Nutzer kennen.

Tracking-Methoden können unterschiedlich gegliedert werden. Eine mögliche Differenzierung ist First-Party-Tracking und Third-Party-Tracking. Als First-Party wird immer die Website bezeichnet, zu der der Nutzer bewusst eine Verbindung aufgebaut hat. Dass diese Website selbst an den Daten des Nutzers interessiert ist, könnte zum Beispiel daran liegen, dass sich die Website über (zielgerichtete) Werbung finanziert. Third-Partys dagegen sind Websites, die Inhalte beisteuern, wenn gleich der Nutzer nicht aktiv eine Verbindung zu diesen aufgebaut hat. Dies funktioniert durch Einbinden einer Third-Party in die Website einer First-Party. Mögliche Gründe hierfür könnten sein, dass die beiden Unternehmen hinter den Websites zusammengehören (z.B. Google und Youtube) oder dass die Third-Party dafür bezahlt, Daten sammeln zu dürfen.

Gerade die genannte Zusammengehörigkeit von Unternehmen spielt auch eine wichtige Rolle beim Tracking. Auf diese Weise können nämlich Daten zusammengeführt werden und noch umfangreichere Profile über einzelne Nutzer erstellt werden. Dies dürfte zum Beispiel auch ein Grund für *Facebook* gewesen sein, 19 Milliarden US-Dollar für die Übernahme von *WhatsApp* aufzubringen, auch wenn Mark Zuckerberg (Gründer und Vorsitzender von Facebook) es damit begründet, die gesamte Welt vernetzen zu wollen. [19]

Grundsätzlich muss Tracking von zwei verschiedenen Seiten betrachtet werden, um einschätzen zu können, ob dies positiv oder negativ ist. Aus Sicht der Webseitenbetreiber hat Tracking durchaus positive Effekte. Daten sind mittlerweile nämlich zu einer Art Zahlungsmittel geworden. Da viele Websites ihre Inhalte für Nutzer kostenlos zur Verfügung stellen, ist schlichtweg eine andere Einnahmequelle erforderlich. Aus Sicht der Nutzer hat Tracking allerdings zumeist einen negativen Beigeschmack. Dies liegt an der Art und Weise, wie Tracking betrieben wird. So wird der Nutzer in der Regel nicht umfassend darüber informiert, welche Daten auf welche Weise erhoben werden und wem diese Informationen zufließen. Diese Seminararbeit hat das Ziel, den Leser über verschiedene Tracking-Verfahren zu informieren und aufzuzeigen, wie man sich gegen solche Verfahren schützen kann. Einen hundertprozentigen Schutz gibt es allerdings nicht (zumindest in den Augen des Autors). Natürlich ist das Thema „Tracking" viel zu umfangreich, um jedes einzelne Verfahren zu beschreiben. Deshalb wurde in dieser Arbeit eine Auswahl der bekanntesten Verfahren (empirische Wahrnehmung des Autors) getroffen.

doi: 10.2313/NET-2015-09-1_04

In Kapitel 2 und 3 werden verschiedene Angriffsmöglichkeiten beschrieben. Kapitel 4 veranschaulicht verschiedene Tracking-Verfahren in Form einer Beispielimplementierung. Kapitel 5 beschäftigt sich anschließend mit verschiedenen Schutzmaßnahmen.

2. ANGRIFFSMÖGLICHKEITEN

2.1 Explizite Verfahren

Als explizite Tracking-Verfahren werden alle Verfahren bezeichnet, bei denen der Tracker Informationen auf dem Computer des Nutzers abspeichert, welche er später zur Wiedererkennung abfragen kann. [1] Explizite Verfahren stellen vermutlich die älteste Art von Tracking dar, weil diese wesentlich weniger komplex sind als schließende Verfahren (siehe Abschnitt 2.2). Hierbei werden im Grunde Mechanismen missbraucht, welche für andere Aspekte konzipiert wurden. Der Nachteil für Tracker besteht allerdings darin, dass diese Verfahren eine eindeutige Spur auf dem Computer des Nutzers hinterlassen. Ein erfahrener Nutzer kann somit die gespeicherten Informationen manipulieren oder ganz löschen.

2.1.1 HTTP-Cookies

Die einfachste Art, Informationen auf dem Computer des Nutzers abzuspeichern, stellen Cookies dar. Ein Cookie ist eine Textdatei, die im Webbrowser gespeichert und zwischen Webserver und Webbrowser ausgetauscht wird. Ein Cookie muss mindestens über einen serverseitig eindeutigen *Namen* verfügen, optional sind auch noch weitere Attribute möglich. Die wichtigsten sind der *Wert* des Cookies (Information), das *Ablaufdatum* und der *Pfad* auf dem Server, für den das Cookie verfügbar ist. Wird kein Ablaufdatum angegeben, existiert das Cookie nur bis zum Ende der Sitzung, also bis der Browser geschlossen wird. Wie viel Speicherplatz für ein Cookie zur Verfügung steht, ist browserabhängig, jedoch schreibt die Spezifikation vor, dass ein Browser mindestens 4096 Bytes zur Verfügung stellen muss. [15]

Der eigentliche Sinn von Cookies besteht darin, verschiedene Einstellungen zu speichern. Zum Beispiel kann die bevorzugte Sprache gespeichert werden, sodass der Webserver bei einem zweiten Besuch diese Information sofort zur Verfügung hat und gleich den „richtigen" Inhalt liefern kann. Allerdings wurde auch schnell erkannt, dass man Cookies dazu nutzen kann, um einen Besucher wieder zu identifizieren, womit man beim Thema Tracking ist. Hierfür muss der Server lediglich jedem Nutzer einen eindeutigen Wert zuweisen, welchen er in einem Cookie speichert. Somit kann der Nutzer zu einem späteren Zeitpunkt unmissverständlich wieder erkannt werden. Es fällt auf, dass die häufigsten vorkommenden Bezeichner *utma, utmb, utmc* und *utmz* heißen. Dies sind bekannte Cookies von *Google Analytics*, welche genau den genannten Zweck erfüllen. [21] Das beweist, dass diese Methode tatsächlich oft eingesetzt wird.

Grundsätzlich kann ein Webserver nur Cookies der eigenen Domain auslesen. Angenommen es existiert auf dem Computer des Nutzers ein Cookie von *www.example-one.com* und ein Cookie von *www.example-two.com*. Ein Dienst, welcher unter *www.example-one.com* läuft, hat also keine Möglichkeit, das Cookie von *www.example-two.com* auszulesen. [15] Gerade für Tracking-Mechanismen ist es aber wichtig, Informationen über Nutzer über die eigenen Websites hinaus zu erlangen, da somit wesentlich mehr Informationen in Erfahrung gebracht und folglich umfangreichere Profile erstellt werden können. Um dies zu bewerkstelligen, werden sogenannte *Third-Party-Cookies* eingesetzt, d.h. Cookies, die von einem anderen Webserver gesetzt werden, als von dem auf dessen Website man gerade ist. Hierzu muss der Webbrowser dazu gebracht werden, eine Verbindung zu der Third-Party aufzubauen, sodass auch diese Cookies setzen kann. Dies geschieht über das Einbinden von externen Elementen und wird im Detail in Abschnitt 3 erläutert.

Cookies sind heutzutage immer noch das meist verwendete Tracking-Verfahren, weil sie einerseits mit sehr geringem Aufwand gesetzt, geändert oder gelöscht werden können und andererseits von jedem Browser unterstützt werden. [2] Außerdem haben die meisten Nutzer nicht das nötige Wissen, mit Third-Party-Cookies umzugehen. Die Nachteile für Tracker sind allerdings das Ablaufdatum und der geringe Speicher.

Nach der EU-Richtlinie 2009/136/EG (Artikel 5 Absatz 3) muss eine Website Nutzer darüber informieren, zu welchem Zweck Cookies gesetzt werden und diese müssen ihre Einwilligung abgeben. [22] In der Regel befinden sich die Bedingungen im Impressum der Website und der Nutzer stimmt implizit durch die Nutzung der Website zu.

2.1.2 HTML5 Storage

HTML5 ist der Nachfolger des bewährten HTML4-Standards. HTML5 bietet neue Features, wie beispielsweise das **<video>**-Element, welches in Zukunft *Flash* ersetzen könnte, oder das **<canvas>**-Element, welches direktes Zeichnen im Browser ermöglicht (siehe dazu auch Abschnitt 2.2.2). Auch wurde mit HTML5 eine neue Speichertechnologie eingeführt, welche sich untergliedern lässt in *Session-Storage* und *Local-Storage* [9]. In dieser Arbeit wird nur *Local-Storage* betrachtet. Dies funktioniert im Grunde ähnlich wie HTTP-Cookies, d.h. auch hier werden Daten im Browser des Nutzers gespeichert. Local-Storage hat allerdings im Vergleich zu Cookies den Vorteil für Tracker, dass der verfügbare Speicher mit fünf Megabyte deutlich höher ist und kein Ablaufdatum vorhanden ist. Gespeicherte Daten sind also so lange verfügbar, bis sie entweder durch die Website oder manuell durch den Nutzer gelöscht werden. [10]

Genau wie bei Cookies kann auch Local-Storage für Tracking eingesetzt werden. Eine Möglichkeit wäre auch hier, eindeutige Identifikationswerte zu speichern, sodass die Website einen Nutzer bei einem zweiten Besuch sofort wieder identifizieren kann. Auch das Einbinden von dritten Parteien kann analog zu Cookies erfolgen. Gerade das fehlende Ablaufdatum macht Local-Storage deutlich attraktiver für Tracking-Zwecke als Cookies.

Im Gegensatz zu Cookies können Local-Storage-Daten allerdings nur clientseitig mittels JavaScript gelesen werden. Die Daten werden auch nicht im HTTP-Header an den Server gesendet. [9] Würde eine Website anstatt Cookies nur Local-Storage einsetzen, hätte dies den Vorteil, dass die Website schneller laden kann, da weniger Daten an den Server gesendet werden. Für Tracker ist dies allerdings eher nachteilig, da der Server somit keinen direkten Zugriff auf die Daten hat. Ein *Workaround*, um die Daten dennoch an den Server zu senden, kann mittels AJAX (Asynchronous JavaScript and

XML) erfolgen. Dies wird auch in der Beispielimplementierung in Abschnitt 4 angewendet.

HTML5 wird mittlerweile von den meisten Browsern unterstützt [10], sodass davon ausgegangen werden kann, dass Tracking-Verfahren mittels Local-Storage in Zukunft zunehmen werden. Dies zeigt auch der Web-Privacy-Zensus von 2014 im Vergleich zu 2012. [2]

2.1.3 Flash-Cookies

Local Shared Objects (LSOs), umgangssprachlich auch *Flash-Cookies* genannt, wurden von der Firma *Adobe* eingeführt, um Daten auf dem Computer des Nutzers zu speichern. [3] Der Name *Flash-Cookies* ist eine Anlehnung an HTTP-Cookies, da diese ähnliche Funktionalitäten bieten. Flash-Cookies können allerdings nur dann gesetzt werden, wenn der Nutzer ein Flash-Plugin für den Browser installiert und aktiviert hat. Da jedoch viele Websites auf Flash-Technologien basieren bzw. basierten, wie beispielsweise das Videoportal *YouTube*, haben fast alle Nutzer - wenn auch nicht immer bewusst - ein entsprechendes Plugin installiert. Zwar verzichtet *YouTube* seit 2015 standardmäßig in den Browsern *Google Chrome, Internet Explorer 11* und *Safari 8* auf Flash und setzt stattdessen HTML5 <video> ein, doch werden alle anderen Browser immer noch über Flash bedient. [23]

Flash-Cookies wurden dafür entwickelt, um Einstellungen wie beispielsweise die Lautstärke zu speichern. Diese Technologie kann aber auch für Tracking-Zwecke eingesetzt werden, indem wiederum eindeutige Identifikationswerte gespeichert werden. Flash-Cookies haben genau wie HTML5 Storage kein Ablaufdatum, d.h. sie sind solange persistent im Speicher, bis sie manuell gelöscht werden. Die Speichergröße liegt mit 100KB zwar deutlich unter dem verfügbaren Speicher von HTML5 Storage, jedoch höher als bei HTTP-Cookies. [3]

Außerdem macht die Tatsache, dass Flash-Cookies nicht an einem browserspezifischen Ort, sondern zentral auf der Festplatte des Nutzers gespeichert werden, diese sehr attraktiv für Tracking. Aus diesem Grund können gespeicherte Daten browserunabhängig verwertet werden. Wenn ein Nutzer also verschiedene Browser verwendet, um im Internet zu surfen, sind seitenübergreifende Tracking-Verfahren, welche auf Flash-Cookies basieren, mächtiger als beispielsweise HTTP-Cookies oder HTML5 Storage.

Die genannte Browserunabhängigkeit ist mit Sicherheit auch ein Grund dafür, dass Flash-Cookies immer wieder mit dem Stichwort *Respawning* in Verbindung gebracht werden. Respawning bezeichnet die Fähigkeit, durch den Nutzer gelöschte Cookies wiederherzustellen. Hierzu muss im Grunde nur eine exakte Kopie des HTTP-Cookies in einem Flash-Cookie gespeichert werden. Auch kann durch Respawning die Browserabhängigkeit von HTTP-Cookies „überwunden" werden. Besucht ein Nutzer beispielsweise eine Website mit Mozilla Firefox und anschließend mit Google Chrome, kann die Website mithilfe von Respawning in Firefox gesetzte Cookies sofort in Chrome übertragen und somit den Nutzer wieder identifizieren. [4]

Insgesamt bieten Flash-Cookies also eine gute Ausgangssituation für Tracking. Dies liegt vermutlich auch daran, dass die meisten Nutzer überhaupt nicht wissen, dass Flash-Cookies auf ihrem Computer gespeichert sind. Der Web-Privacy-Zensus [2] zeigt allerdings, dass Flash-Cookies seit 2012 weniger geworden sind. Dies liegt daran, dass Flash aufgrund von Sicherheitslücken immer wieder in der Kritik steht und seit der Einführung von HTML5 durch das **<video>**-Element eine einfache Alternative zu Flash zur Verfügung steht. [10] Der Streaming-Dienst *Netflix* zum Beispiel unterstützt HTML5 offiziell zumindest bereits für den Internet Explorer und Google Chrome für Windows und Safari und Chrome für Mac OS X. [17]

2.1.4 Evercookies

Evercookies sind keine neue Technologie, sondern lediglich eine Kombination aus den bereits genannten Verfahren und noch weiteren. Aus Sicht eines Trackers ist es wünschenswert, dass Daten persistent auf dem Computer eines Nutzers gespeichert werden. Dazu zählt, dass die Daten kein Ablaufdatum haben und sie nicht vom Nutzer gelöscht werden können. Dies wird theoretisch durch einen Evercookie erreicht. [16] Nutzer mit viel Hintergrundwissen können allerdings trotzdem in der Lage sein, einen Evercookie zu löschen.

Das Ziel eines Evercookies ist es, sich an so vielen verschiedenen Stellen wie möglich im Computer zu manifestieren, sodass der Nutzer den Überblick verliert. Kombiniert wird die redundante Speicherung anschließend mit Respawning-Techniken (siehe dazu Abschnitt Flash-Cookies 2.1.3). Eine JavaScript-basierte API, welche einen Evercookie erzeugt, wurde von Samy Kamkar entwickelt. Laut Angaben auf seiner Homepage nutzt ein Evercookie bis zu 16 verschiedene Technologien, um sich redundant auf dem Computer des Nutzers zu speichern, darunter HTTP-Cookies, Flash-Cookies, HTML5 Storage, ETags und Silverlight-Cookies, wobei die beiden letzten Verfahren in dieser Arbeit nicht behandelt werden. [16]

Aufgrund der persistenten Speicherung bieten Evercookies somit die mit Abstand besten Voraussetzungen für explizite Tracking-Verfahren. Es ist allerdings unklar, wie viele Websites tatsächlich Evercookies einsetzen.

2.2 Schließende Verfahren

Als schließende Tracking-Verfahren werden alle Verfahren bezeichnet, bei denen der Tracker versucht, einen Nutzer anhand seiner Konfigurations- und Systemeigenschaften zu identifizieren. Schließende Verfahren zählen als fortgeschrittene Tracking-Verfahren, da hierbei im Gegensatz zu expliziten Verfahren (siehe Abschnitt 2.1) keine Spur auf dem Computer des Nutzers hinterlassen wird. [1] Dies macht es für den Nutzer bedeutend schwieriger, sich gegen Tracking zu schützen.

2.2.1 Klassisches Fingerprinting

Das erste Beispiel eines schließenden Verfahrens beschreibt klassisches Fingerprinting. Hierbei versucht die Tracking-Website, meistens mithilfe von JavaScript Eigenschaften des Browsers bzw. des Computers in Erfahrung zu bringen. Die gesammelten Daten können anschließend in einer Datenbank gespeichert und in Zukunft dazu verwendet werden, um einen Nutzer bei einem zweiten Besuch wieder zu identifizieren. Da der durchschnittliche Nutzer nur selten seine

Konfigurationseigenschaften ändert, ist dies auch ein sehr zuverlässiger Tracking-Mechanismus. [5]
Die Beispielimplementierung in Abschnitt 4 zeigt außerdem, dass der Nutzer keinen Einfluss darauf hat, an welche dritten Parteien die gesammelten Daten weitergegeben werden, wenn er nicht gerade aktiv versucht, die Erhebung der Daten zu verhindern.

Das Projekt *Panopticlick* von der Electronic Frontier Foundation [7] hat gezeigt, dass bereits wenige Eigenschaften ausreichend sind, um einen Nutzer eindeutig identifizieren zu können. Zu diesen Eigenschaften gehören die Zeitzone, die Bildschirmauflösung und Farbtiefe, die Liste der installierten Browser-Plugins und System Fonts (Schriftarten), sowie der User-Agent, welcher standardmäßig ohnehin im HTTP-Header an den Server gesendet wird.

Viele der genannten Eigenschaften können bereits ohne Aufwand mithilfe des JavaScript-Objekts *navigator* ausgelesen werden. Lediglich die Liste der installierten Browser-Plugins und System Fonts benötigen tiefgreifendere Techniken.

Klassisches Fingerprinting kann außerdem dazu verwendet werden, einen Nutzer über verschiedene Endgeräte hinweg zu identifizieren. Dies ist allerdings aus Tracker-Sicht gesehen sehr schwierig und setzt Tracking über einen langen Zeitraum voraus. Hierbei wird nämlich davon ausgegangen, dass unterschiedliche Nutzer unterschiedliche Gewohnheiten bezüglich Surfverhalten und Konfigurationen haben, was diese eindeutig identifizierbar macht. Erkennt man beispielsweise, dass ein Nutzer immer wieder die gleichen Websites aufruft (möglicherweise auch in der gleichen Reihenfolge), kann man nach langer Beobachtung darauf schließen, dass es sich um den selben Nutzer handeln muss und dieser lediglich verschiedene Endgeräte verwendet. [1]

2.2.2 Canvas-Fingerprinting

Ein relativ neues Verfahren des Fingerprintings beschreibt das sogenannte Canvas-Fingerprinting. Das **<canvas>**-Element wurde mit HTML5 eingeführt und ermöglicht direktes Zeichnen im Browser mittels JavaScript. Canvas-Fingerprinting bietet ein hohes Potential für Tracking, für welches es bis heute (Stand 2015) keine optimalen Gegenmaßnahmen gibt. [6] (Quelle von 2012, seitdem allerdings keine neuen brauchbaren Verfahren bekannt)

Beim Canvas-Fingerprinting wird in der Regel ein beliebiger Text mithilfe des **<canvas>**-Elements im Browser des Nutzers gezeichnet. Der Fingerprint ergibt sich dadurch, dass jeder Browser aufgrund von Soft- und Hardwarekonfigurationen den Text unterschiedlich darstellt. So spielen beispielsweise das verwendete Betriebssystem, die Liste der installierten System Fonts, die Grafikkarte oder auch die Grafikkartentreiber eine Rolle. [6] Das entstandene Bild wird anschließend mit einer Hash-Funktion in eine binäre Darstellung verwandelt und an den Server zur Verwertung geschickt.

Mowery und Shacham berichteten in ihrer Arbeit *Pixel Perfect: Fingerprinting Canvas in HTML5* zum ersten mal über Canvas-Fingerprinting. Die beiden Autoren haben gezeigt, dass Unterschiede in der Darstellung sogar mit bloßem Auge deutlich erkennbar sind. Spätestens aber nach dem Hashen in eine binäre Darstellung ergeben sich eindeutige Unterscheidungen. [6] Nach empirischer Wahrnehmung ändern sich die Systemeigenschaften beim durchschnittlichen Nutzer nur selten. Dies zeigt, dass Canvas-Fingerprinting ein effektives Verfahren ist, Nutzer eindeutig wieder zu identifizieren.

Für den Fingerprint hat das gezeichnete Bild überhaupt keine Bedeutung, sondern lediglich die binäre Darstellung. Deshalb wird Canvas-Fingerprinting meistens im Hintergrund und für den Nutzer nicht sichtbar vollzogen. Dies geschieht im Bruchteil einer Sekunde.
2014 betrieben circa 5,5% der am besten bewerteten Websites aktives Canvas-Fingerprinting. [4]

2.2.3 History-Sniffing

History-Sniffing beschäftigt sich in erster Linie nicht damit, einen Nutzer wieder zu identifizieren, sondern damit, weitere Informationen über einen Nutzer zu erfahren. Konkret geht es darum, in Erfahrung zu bringen, welche Websites ein Nutzer bereits besucht hat. Dies gibt nämlich Aufschluss über die Interessen des Nutzers und ist somit von großer Bedeutung für Tracker.

Die meisten Browser speichern in der Standardeinstellung einen Verlauf über sämtliche besuchte Websites. Diese Einstellung wird auch von den meisten Nutzern nicht geändert. Von Sniffing-Verfahren, welche Sicherheitslücken ausnützen, um den Verlauf des Browser auszulesen, soll in dieser Arbeit abgesehen werden. Stattdessen soll ein anderes Verfahren beschrieben werden, welches allerdings aufgrund der Reaktion der Browserhersteller so nicht mehr funktioniert. Das Verfahren basierte darauf, dass Browser besuchte Hyperlinks in einer anderen Farbe darstellen, als nicht besuchte. Mithilfe von JavaScript war es möglich, die Farbe eines Hyperlinks zu ermitteln und daraus zu schließen, welche Seiten der Nutzer bereits besucht hat. [1] Browserhersteller haben aus datenschutzrechtlichen Gründen darauf reagiert. In Mozilla Firefox funktioniert beispielsweise die Funktion *getComputedStyle* nicht mehr, sodass die Methode keine wirkliche Relevanz mehr hat. [8]

Das Verfahren zeigt allerdings, wie kreativ Tracker sind und wie mächtig JavaScript ist, sodass mit Sicherheit davon ausgegangen werden kann, dass in Zukunft wieder ähnliche Verfahren entwickelt werden.

3. EINBINDEN VON DRITTEN PARTEIEN

Unter „Einbinden von dritten Parteien" versteht man, dass ein Webseitenbetreiber im Hintergrund Elemente eines Trackers einbindet und somit eine Verbindung zu diesem herstellt. Das Ziel dabei ist es, dritten Parteien die Möglichkeit zu bieten, Tracking zu betreiben. Auf diese Weise können auch dritte Parteien explizite oder schließende Tracking-Verfahren anwenden. [1]

In vielen Fällen wird das Einbinden externer Elemente überhaupt nicht verschleiert, sondern direkt durch sichtbare Elemente durchgeführt. Den meisten Nutzern ist allerdings nicht bewusst, dass hierdurch auch Tracking betrieben wird.

Das technische Werkzeug hierfür sind *I-Frames*, welche mit HTML4 eingeführt wurden. Hierbei handelt es sich um ein

doi: 10.2313/NET-2015-09-1_04

Element, welches das parallele Laden aus verschiedenen Quellen ermöglicht. [1] Beispielsweise stammen Grafiken oder Videos oft nicht direkt von der Website, sondern von dritten Parteien, wie *YouTube*. Auch befinden sich auf vielen Websites Wegbeschreibungen in Form von eingebundenen *Google Maps*-Landkarten. Neben Tracking-Zwecken hat dies auch den einfachen Grund, um Speicherplatz einzusparen, da die Inhalte somit nicht redundant auf jedem einzelnen Server gespeichert werden müssen.

Auf vielen Websites befinden sich außerdem sogenannte *Social-Plugins* von *Facebook, Twitter* und *Google+*. Diese Plugins bieten noch bessere Tracking-Möglichkeiten, da hiermit besuchte Websites eines Nutzers in Erfahrung gebracht werden können und diese Daten mit der echten Identität des Nutzers verknüpft werden können. [1] Voraussetzung hierfür ist natürlich, dass der Nutzer auf einer dieser sozialen Plattformen registriert ist und hierbei seine echten Daten angegeben hat. Bei 1,39 Milliarden Facebook-Nutzern (Stand: 2015) [18] ist die Wahrscheinlichkeit hierfür allerdings sehr hoch. Dies zeigt, wie effektiv diese Tracking-Methode ist.

Die zweite Art, Elemente eines Trackers einzubinden, sind unsichtbare Elemente. Hierfür kann zum Beispiel ein PHP-Skript einer dritten Partei in die Website eingebunden werden. Dies wird auch in der Beispielimplementierung in Abschnitt 4 gezeigt. Da PHP serverseitig ausgewertet wird, hat der Nutzer somit auch keine Möglichkeit, zu erkennen, welche Daten an das eingebundene PHP-Skript übergeben werden.

4. BEISPIELIMPLEMENTIERUNG

Im Folgenden werden einige der genannten Tracking-Techniken in einer Beispielimplementierung kombiniert. Konkret werden HTML5 Storage, Fingerprinting und die Einbindung eines externen PHP-Skripts verwendet. Das Beispiel wurde in einem lokalen Netzwerk getestet. Der Aufbau wird in Abbildung 1 verdeutlicht.

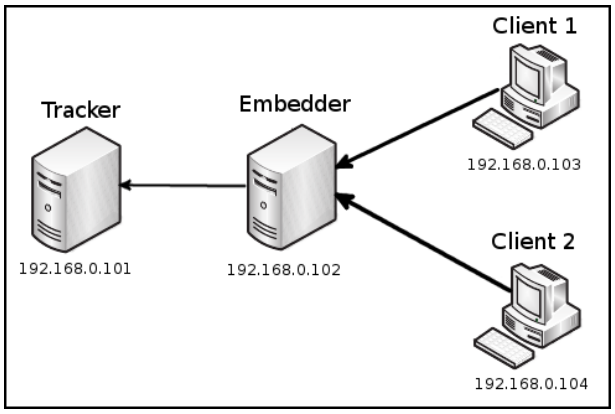

Abbildung 1: Aufbau zur Implementierung

Abbildung 1 zeigt, dass der Embedder-Server als First-Party eine Website anbietet, auf welche zwei verschiedene Clients zugreifen. Im Hintergrund wird eine Verbindung zu dem Tracker-Server aufgebaut und diesem verschiedene Daten gesendet. Der Tracker-Server wiederum kann die empfangenen Daten anschließend in einer Datenbank speichern. Er agiert hier also ausschließlich als Third-Party.

Listing 1 zeigt einen Ausschnitt aus *index.php* des Embedder-Servers. Die Zeilen 16 bis 19 prüfen, ob der Nutzer die Website zum ersten Mal besucht. Ist dies der Fall (oder hat der Nutzer Local-Storage zurückgesetzt), wird eine Local-Storage Variable namens „counter" angelegt und diese bei jedem neuen Besuch ausgelesen und um eins inkrementiert.

In den Zeilen 21 und 22 wird Fingerprinting betrieben. Da dieses Beispiel nur zur Veranschaulichung dienen soll, werden hier lediglich die Bildschirmauflösung und der User-Agent ausgelesen. Das Paper *User Tracking on the Web via Cross-Browser Fingerprinting* [5] hat allerdings gezeigt, dass alleine der User-Agent schon in etwa 50 Prozent der Fälle ein eindeutiges Erkennungsmerkmal ist.

Die Zeilen 24 bis 26 lösen das Problem, die mit JavaScript ausgelesenen Variablen an PHP zu übergeben. Hierfür wird AJAX (Asynchronous JavaScript and XML) verwendet, d.h. die Daten werden in einem XMLHttpRequest verpackt über *GET-Variablen* zurück an den Index gesendet.

Im PHP-Teil (Zeilen 1 bis 8) können diese GET-Variablen nun verwendet werden. Zusätzlich wird noch die IP-Adresse des Nutzers ausgelesen. Alle diese Daten (IP-Adresse, Anzahl der Besuche, Bildschirmauflösung und User-Agent) werden nun an den Tracker übergeben (wieder über GET-Variablen). Hierzu wird lediglich ein PHP-Skript des Trackers mithilfe des *file*-Befehls eingebunden.

Auf der Seite des Tracker-Servers können die empfangenen Daten nun ausgelesen und in einer Datenbank gespeichert werden. Existiert eine IP-Adresse bereits in der Datenbank, wird lediglich die Anzahl der Besuche aktualisiert. Natürlich könnte der Tracker auch zusätzlich als First-Party eine eigene Website anbieten und somit selbst Informationen über die Besucher ermitteln. Diese Daten können anschließend mit denen aus der Datenbank abgeglichen werden, um auf diese Weise umfangreichere Erkenntnisse über die Nutzer zu erhalten.

```
1  <?php
2    $ip = $_SERVER['REMOTE_ADDR'];
3    $count = $_GET['count'];
4    $resolution = $_GET['resolution'];
5    $userAgent = $_GET['userAgent'];
6
7    @file('http://192.168.0.101/tracker/index.php
         ?ip='.$ip.'&count='.$count.'&resolution='
         .$resolution.'&userAgent='.urlencode(
         $userAgent));
8  ?>
9
10 [...]
11
12 You have visited this page <span id="count"></
       span> times.
13
14 <script type="text/javascript">
15
16   if(!localStorage['counter'])
17     localStorage['counter'] = 0;
18   localStorage['counter'] = parseInt(
         localStorage['counter']) + 1;
19   document.getElementById('count').textContent
         = localStorage['counter'];
20
```


doi: 10.2313/NET-2015-09-1_04

```
21    var resolution=[screen.width,screen.height];
22    var userAgent = navigator.userAgent;
23
24    xmlhttp = new XMLHttpRequest();
25    xmlhttp.open('GET','index.php?count='+
          localStorage['counter']+'&resolution='+
          resolution+'&userAgent='+userAgent, true)
          ;
26    xmlhttp.send();
27
28  </script>
```

Listing 1: Auszug aus Embedder: Index.php

Das Beispiel verdeutlicht, mit welch einfachen Mitteln Tracking möglich ist und wie „machtlos" der Nutzer dagegen ist. Schaut sich der Nutzer beispielsweise den Seitenquelltext der Website in seinem Browser an, könnte er lediglich noch erkennen, dass hier verdächtige Werte an den Server zurückgesendet werden. Was der PHP-Teil mit den Daten allerdings macht, ist für den Nutzer nicht erkennbar, da PHP serverseitig ausgewertet wird und somit überhaupt nicht im Seitenquelltext erscheint. Zu beachten ist allerdings, dass die Implementierung wirklich nur zur Veranschaulichung dienen soll und bewusst einfach gehalten wurde. Es wurde beispielsweise kein Fokus auf Sicherheitsaspekte (z.B. SQL-Injection) gelegt. Auch sollte in einem echten Anwendungsfall nicht der *file*-Befehl verwendet werden. Dies hat nämlich den Nachteil, dass die Website auf die Antwort des Tracker-Servers warten wird. Ist der Tracker-Server nicht erreichbar, wird die komplette Website nicht laden können.

5. SCHUTZMAßNAHMEN

Die vorangegangenen Abschnitte haben sowohl gezeigt, wie viele Tracking-Arten es gibt, als auch dass Personen, welche Tracking betreiben, immer kreativer werden. Das beste Beispiel hierfür ist *Canvas-Fingerprinting* (siehe Abschnitt 2.2.2). Es verlangt einiges an Kreativität, ein Tracking-Verfahren zu konstruieren, welches auf einem Element zum direkten Zeichnen im Browser basiert.

Aus diesen Gründen kann Tracking wohl kaum komplett verhindert werden. Dennoch gibt es Möglichkeiten für den Nutzer, Tracking zumindest einzuschränken.

5.1 Strategie: Blockieren

Die einfachste und zugleich auch effektivste Art, etwas zu verhindern, besteht darin, es komplett zu blockieren.

5.1.1 Explizite Verfahren

Alle guten Browser bieten die Möglichkeit, *Third-Party-Cookies* zu blockieren. Die Standard-Einstellung lässt in den meisten Fällen allerdings Third-Party-Cookies zu, sodass dies bereits ein technisches Wissen des Nutzers voraussetzt. In der Regel hat der Nutzer durch das Blockieren keinen Nachteil, da Third-Party-Cookies meistens für Tracking-Zwecke eingesetzt werden. [1] Gelegentlich kann es aber auch zu Funktionseinbußen kommen. Beispielsweise werden Third-Party-Cookies benötigt, um Kommentare zu YouTube-Videos abzugeben. Dies liegt daran, dass das Kommentieren bei YouTube über einen *Google+*-Account abgehandelt wird, wobei *Google+* in diesem Fall dann die Rolle einer Third-Party einnimmt und deshalb Third-Party-Cookies erfordert. [24]

Dass First-Party-Cookies auch für Tracking-Zwecke eingesetzt werden können, wurde bereits in Abschnitt 2.1.1 erklärt. Dies zu verhindern, ist nicht so einfach möglich. Zwar bieten die meisten Browser auch die Möglichkeit, First-Party-Cookies zu blockieren, doch funktionieren dadurch einige Websites nicht mehr richtig. Beispielsweise setzen *Facebook* oder *Amazon* Cookies voraus, um sich überhaupt anmelden zu können. Eine Kompromisslösung, welche allerdings einen höheren Verwaltungsaufwand nach sich zieht, ist das Verwenden einer *White-List*. Das bedeutet, man erlaubt das Setzen von Cookies nur den Websites, denen man vertraut bzw. die ohne Cookies nicht funktionieren. Alle anderen Websites werden daran gehindert, Cookies zu setzen.

Das Verwenden einer White-List (oder auch einer Black-List) kann für einen Nutzer allerdings problematisch sein in Bezug auf klassisches Fingerprinting. Das individuelle Anlegen dieser Listen kann ausschlaggebend dafür sein, dass man eindeutig wieder identifiziert wird. Schützt man sich also gegen ein bestimmtes Verfahren, macht man sich gleichzeitig angreifbar für ein anderes Verfahren. Dies zeigt ein weiteres Mal, dass Tracking nicht komplett verhindert werden kann.

In Mozilla Firefox ist durch das Blockieren von Cookies auch HTML5 Storage betroffen. Für den Umgang mit Flash-Cookies benötigt es allerdings einen lokalen Einstellungsmanager für den Flash-Player [1], da Flash-Cookies browserunabhängig gespeichert werden.

Ein weiterer Ansatz zum Verhindern von expliziten Verfahren, ist die Verwendung des *Private-Modes* (teilweise Inkognito-Mode genannt). Dies wird von allen guten Browsern angeboten, darunter *Mozilla Firefox, Google Chrome, Safari* oder auch der *Internet-Explorer*. Durch diesen Modus werden keine Informationen auf dem Computer des Nutzers dauerhaft gespeichert. [25] Cookies, HTML5-Storage und ähnliches sind nur temporär verfügbar und werden nach Beenden des Modus gelöscht. Auf diese Weise erfährt der Nutzer keine Funktionseinbußen, obwohl explizites Tracking weitestgehend unterbunden wird. Der *Private-Mode* bietet allerdings keinen Schutz gegen schließende Verfahren.

5.1.2 Schließende Verfahren

Schließende Tracking-Verfahren, wie klassisches Fingerprinting oder Canvas-Fingerprinting, beruhen in der Regel immer auf JavaScript. Mit der Erweiterung *NoScript* [11] für Mozilla Firefox ist es möglich, JavaScript komplett zu blockieren. Dies verhindert die genannten Tracking-Verfahren. Allerdings gibt es heutzutage fast keine Websites mehr, die ohne JavaScript richtig funktionieren. Somit ist diese Methode eher ein theoretischer, als praktisch anwendbarer Schutz. Als Kompromiss kann auch hier wieder eine White-List verwendet werden.

Ein weiteres Addon für Mozilla Firefox, welches angeblich Canvas-Fingerprinting verhindern soll, heißt *CanvasBlocker* [12] und wurde im Zuge dieser Seminararbeit genauer getestet. Obwohl das Addon überwiegend positive Bewertungen erhalten hat, war das Ergebnis der Tests nur mäßig zufriedenstellend. Dies verdeutlicht ein weiteres Mal, dass es gegen Canvas-Fingerprinting bis heute (Stand: 2015) keine wirklich

brauchbare Gegenmaßnahme gibt.
Das Tool wurde in der Einstellung „um Erlaubnis fragen" getestet. Auf vielen Websites, darunter *bundesliga.de, kawasaki.de* und *audi.de*, wurden versteckte Canvas-Elemente gefunden und diese nach Nachfrage blockiert. Ein Unterschied der Funktionalität dieser Seiten mit aktiviertem oder deaktiviertem CanvasBlocker konnte nicht festgestellt werden. Dies deutet stark auf Canvas-Fingerprinting hin, muss aber nicht zweifellos der Fall sein. Beispielsweise wurde auf der Website *maps.google.de* auch ein verstecktes Canvas-Element gefunden. Es stellte sich jedoch heraus, dass dieses versteckte Element für die Suchmaske verantwortlich ist und die Website somit mit aktiven CanvasBlocker unbrauchbar ist (abgesehen von der Verwendung einer White-List).

```
1 <canvas id="can" width="100" height="100">
2 </canvas>
3
4 <script type="text/javascript">
5   var canvas = document.getElementById("can");
6   var context = canvas.getContext("2d");
7   context.fillRect(0,0,100,100);
8 </script>
```

Listing 2: JavaScript Code für ein Quadrat

Listing 2 zeigt einen JavaScript Code, welcher ein 100x100 Pixel großes Quadrat zeichnen sollte. Aus einem unersichtlichen Grund wurde der Code mit aktiviertem CanvasBlocker ohne Nachfrage komplett blockiert, obwohl die Einstellung „um Erlaubnis fragen" aktiviert war. Dies zeigt deutlich, dass das Tool mit Vorsicht zu genießen ist und nicht optimal gegen Canvas-Fingerprinting schützt.

Auch der *Tor Browser* versucht aktiv, Canvas-Fingerprinting zu verhindern. Die verwendete Strategie ist die gleiche wie beim *CanvasBlocker*. Der Browser frägt nach, ob ein <canvas>-Element eingebunden werden darf, falls eine Website ein solches verwenden möchte. [26] Der Tor Browser wurde allerdings in dieser Seminararbeit nicht getestet.

Eines der beliebtesten Browser-Addons überhaupt ist *Adblock-Plus* [13], welches für diverse Browser verfügbar ist. Das Addon hat hauptsächlich die Aufgabe, in Websites integrierte Werbungen zu blockieren. Technisch realisiert wird dies über verschiedene *Black-Lists*. Das bedeutet, dass die Website nach Elementen durchsucht wird, welche sich in einer der verwendeten Black-Lists befinden und diese daraufhin blockiert werden. *Adblock-Plus* kann aber auch als Schutz gegen Tracking betrachtet werden. Zum einen existiert eine Black-List[1], welche Social-Plugins blockiert, zum anderen wird Werbung blockiert, welche auch für Tracking eingesetzt werden kann. Dies liegt daran, dass Werbung in der Regel von dritten Parteien eingebunden wird und somit Third-Party-Tracking betrieben werden kann (siehe Abschnitt 3), was dadurch unterbunden wird.

5.2 Strategie: Ergebnisse verfälschen

Eine zweite Strategie ist es, Tracking grundsätzlich nicht zu blockieren, sondern unbrauchbare Ergebnisse zu liefern.

[1]https://easylist-downloads.adblockplus.org/fanboy-social.txt

Auf diese Weise haben die durch Tracking erhobenen Daten keine Bedeutung. Inwieweit diese Technik realisiert werden kann und brauchbar ist, konnte im Zuge dieser Seminararbeit nicht geklärt werden. Dies wird für zukünftige Arbeit offen gelassen. Dass es technisch möglich sein muss, kann am Beispiel eines *User Agent Overriders* [14] gezeigt werden. Hiermit ist es beispielsweise möglich, auf einem Linux-System mit Firefox, den User-Agent eines Windows-Systems mit Internet-Explorer anzunehmen. Auf ähnliche Weise sollte es möglich sein, durch JavaScript ausgelesene Daten zu verfälschen. Hieran schließt sich allerdings die Frage an, wie unterschieden werden soll, wann echte und wann verfälschte Ergebnisse geliefert werden. Natürlich haben die ausgelesenen Daten neben Tracking auch sinnvolle Zwecke. Beispielsweise kann die Bildschirmauflösung dazu verwendet werden, um die Website optimal darzustellen und sollte aus diesen Gründen nicht manipuliert werden.

Eine weitere Methode, welche auf der gleichen Strategie basiert, ist die Verwendung eines Anonymisierungsproxys. Auch hier werden Ergebnisse verfälscht bzw. verschleiert, in diesem Fall die eigene IP-Adresse. Bei der Verwendung eines Proxys wird eine Anfrage an einen Server nicht direkt ausgeführt, sondern über den Proxy umgeleitet. Das bedeutet, dass der Proxy als Stellvertreter die Anfrage an den Server stellt und die Antwort an den eigentlichen Nutzer sendet. [1] IP-basierten Tracking-Verfahren kann auf diese Weise entgegengewirkt werden, da die eigene IP-Adresse dem Server nicht bekannt ist.

6. VERWANDTE ARBEITEN

Diese Arbeit hat in erster Linie das Ziel, einen Gesamtüberblick über Tracking und Schutzmaßnahmen zu vermitteln. Deshalb werden hier viele verschiedene Verfahren erläutert. Natürlich kann auf diese Weise aber nicht jedes einzelne Verfahren im Detail erklärt werden. An dieser Stelle soll auf andere Arbeiten verwiesen werden. Beispielsweise werden in *Pixel Perfect: Fingerprinting Canvas in HTML5* von K. Mowery und H. Shacham [6] viele Details über Canvas-Fingerprinting erklärt. Auch die Arbeit *User Tracking on the Web via Cross-Browser Fingerprinting* [5] beschäftigt sich mit dem Thema *Fingerprinting*. Die vier Autoren haben ein ähnliches Projekt wie *Panopticlick* [7] entwickelt und interessante Statistiken dazu erstellt. Für mehr Details im Bereich expliziter Tracking-Verfahren kann die Arbeit *The Web Never Forgets: Persistent Tracking Mechanisms in the Wild* [4] herangezogen werden. Hier werden Aspekte wie Cookie-Syncing, Evercookies und Flash behandelt.

7. ZUSAMMENFASSUNG

Web-Tracking ist eine allgegenwärtig eingesetzte Technik, um Informationen und Interessen eines Nutzers in Erfahrung zu bringen. Ziel ist es unter anderem, einen Nutzer bei einem zweiten Besuch einer Website wieder identifizieren zu können.

Explizite Verfahren sind Verfahren, bei denen der Tracker Informationen auf dem Computer des Nutzers abspeichert. Bei den gespeicherten Werten handelt es sich meistens um eindeutige Identifikationswerte. Dies kann in Form von HTTP-Cookies, HTML5 Storage, Flash-Cookies oder sogenannten Evercookies geschehen.

Schließende Verfahren sind Verfahren, bei denen der Tracker einen Nutzer anhand von Konfigurations- und Systemeigenschaften wieder identifizieren kann. Beim klassischen Fingerprinting werden mittels JavaScript verschiedene Werte ausgelesen, um den Nutzer eindeutig zu identifizieren. Dazu gehören der User-Agent, die Liste der installierten Browser-Plugins sowie System Fonts, die Bildschirmauflösung oder die Zeitzone. Beim Canvas-Fingerprinting wird dagegen ein eindeutiger Fingerprint mithilfe des HTML5 Canvas-Elements erstellt. Dies ist möglich, da jeder Browser aufgrund von Hard- und Softwarekonfigurationen Text in einem Canvas-Element anders darstellt und somit eindeutig ist.

Unter „Einbinden von dritten Parteien" versteht man, dass ein Webseitenbetreiber Elemente eines Trackers einbindet und somit eine Verbindung zu diesem herstellt. Ziel hierbei ist es, dritten Parteien Tracking zu ermöglichen. Dies kann in Form von Grafiken, Videos oder auch PHP-Skripten erfolgen.

Um sich gegen Tracking zu schützen, gibt es zwei verschiedene Strategien. Bei der ersten Strategie versucht man, Tracking komplett zu blockieren. Dies kann zum Beispiel durch statisches *Black-Listing* oder *White-Listing* geschehen. Bei der zweiten Strategie versucht man, unbrauchbare Daten an Tracker zu übergeben, sodass Tracking quasi keine Bedeutung hat. Dies kann zum Beispiel durch die Verwendung von Anonymisierungsproxys erreicht werden.

Da Personen, die Tracking betreiben allerdings sehr kreativ sind und zudem sehr viele verschiedene Tracking-Verfahren existieren, wird es nicht möglich sein, Tracking komplett zu verhindern, ohne dass dabei die Funktionalität der Websites leidet.

8. LITERATUR

[1] M. Schneider, M. Enzmann, M Stopczynski, *Web-Tracking-Report 2014*, 2014, FRAUNHOFER VERLAG

[2] I. Altaweel, J. Cabrera, H. Choi, K. Ho, N. Good, C. Hoofnagle, *Web Privacy Census: HTML5 Storage Takes the Spotlight As Flash Returns*, 2014

[3] M. Ayenson, D. Wambach, A. Soltani, N. Good, C. Hoofnagle, *Flash Cookies and Privacy II*, 2011

[4] G. Acar, E. Eubank, S. Englehardt, M. Juarez, A. Narayanan, C. Diaz, *The Web Never Forgets: Persistent Tracking Mechanisms in the Wild*, 2014

[5] K. Boda, A. Földes, G. Gulyás, S. Imre, *User Tracking on the Web via Cross-Browser Fingerprinting*, 2012, Budapest University of Technology and Economics

[6] K. Mowery, H. Shacham, *Pixel Perfect: Fingerprinting Canvas in HTML5*, 2012, University of California

[7] *Panopticlick: How Unique - and Trackable - Is Your Browser*, `https://panopticlick.eff.org` - last visited: 5. Mai 2015

[8] *Privacy-related Changes coming to CSS:visited*, `https://hacks.mozilla.org/2010/03/privacy-related-changes-coming-to-css-vistited/` - last visited: 5. Mai 2015

[9] *Web Storage*, `http://www.w3.org/TR/webstorage/` - last visited: 5. Mai 2015

[10] *Dive Into HTML5*, `http://diveintohtml5.info/storage.html` - last visited: 5. Mai 2015

[11] *NoScript*, `https://noscript.net/` - last visited: 5. Mai 2015

[12] *CanvasBlocker*, https://addons.mozilla.org/de/firefox/addon/canvasblocker/ - last visited: 5. Mai 2015

[13] *Adblock Plus: Für ein Web ohne nervige Werbung*, `https://adblockplus.org/de/` - last visited: 5. Mai 2015

[14] *User Agent Overrider*, `https://addons.mozilla.org/de/firefox/addon/user-agent-overrider/` - last visited: 5. Mai 2015

[15] *HTTP State Management Mechanism*, `http://tools.ietf.org/html/rfc6265` - last visited: 5. Mai 2015

[16] *Evercookie - never forget*, `http://samy.pl/evercookie/` - last visited: 5. Mai 2015

[17] *Netflix-Systemvoraussetzungen*, `https://help.netflix.com/de/node/23742` -last visited: 5. Mai 2015

[18] *Börsenbericht: Die ersten offiziellen Facebook-Nutzerzahlen im Jahr 2015*, `http://allfacebook.de/zahlen_fakten/facebook-nutzerzahlen-2015` - last visited: 5. Mai 2015

[19] *Facebook übernimmt WhatsApp*, `https://www.tagesschau.de/wirtschaft/facebook460.html` - last visited: 5. Mai 2015

[20] *Gabler Wirtschaftslexikon*, http://wirtschaftslexikon.gabler.de/Definition/datenschutz.html - last visited: 5. Mai 2015

[21] *Google Analytics Cookie Usage on Websites*, https://developers.google.com/analytics/devguides/collection/analyticsjs/cookie-usage - last visited: 5. Mai 2015

[22] *Amtsblatt der Europäischen Union*, http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2009:337:0011:0036:de:PDF - last visited: 5. Mai 2015

[23] *YouTube now defaults to HTML5 <video>*, `http://youtube-eng.blogspot.de/2015/01/youtube-now-defaults-to-html5_27.html` - last visited: 5. Mai 2015

[24] *Commenting in Youtube*, `http://superuser.com/questions/677942/how-do-i-allow-commenting-in-youtube-through-google-if-i-have-third-party-coo` - last visited: 5. Mai 2015

[25] *Privater Modus - Kontrolle über die von Firefox gespeicherten Daten behalten*, `https://support.mozilla.org/de/kb/privater-modus` - last visited: 5. Mai 2015

[26] *The Design and Implementation of the Tor Browser*, `https://www.torproject.org/projects/torbrowser/design/` - last visited: 5. Mai 2015