SITSEC - der Webauftritt durchleuchtet: das privatere, datensparsamere, nachhaltigere und IT-sicherere Web

Bernhard Birnbaum, Meryem Lasri, Jonas Morawietz

Zusammenfassung—Im Zentrum dieses Projekts steht die Fragestellung, inwiefern Drittparteien in die Kommunikation mit Webseiten aus dem Bereich Gesundheit eingebunden sind. Dazu werden Webauftritte mit verschiedenen OpenSource-Werkzeugen mit Fokus auf Datenschutz und -sparsamkeit sowie IT-Sicherheit untersucht. Abschließend sollen geeignete Gegenmaßnahmen evaluiert werden.

Index Terms—IT-Security, Privacy, Health, Open Source

1 Motivation

Webauftritte im Bereich Soziales/Gesundheit sind oftmals alles andere als datensparsam. Es werden Drittparteien in die Kommunikation eingebunden, ungewollt Werbung platziert und datenschutzrelevante Nutzerdaten (nicht zweckgebunden) sowohl an den kontaktierten Server sowie Drittparteien übermittelt [1]. Dabei können zusätzlich IT-sicherheitsrelevante Schwachstellen Nutzende gefährden.
Vor allem ist dabei der Sicherheitsaspekt der Vertraulichkeit gefährdet, da sehr sensible Daten an Drittanbieter weitergegeben werden. Unter Umständen kann allerdings auch die Authentizität sowie Integrität verletzt werden, wenn bsp. die Subresource Integrity nicht korrekt implementiert wurde oder HSTS nicht aktiv ist.
Deshalb sollen Webauftritte mit OpenSource-Werkzeugen in Bezug auf Datensparsamkeit, Datenschutz, Nachhaltigkeit im Sinne des Resourcenverbrauchs sowie IT-Sicherheit untersucht werden. Zusätzlich sollen die Untersuchungsmethoden des *Website Evidence Collectors* um die Funktionalität des Überschreitens von Cookie-Bannern erweitert werden, um realitätsnähere Ergebnisse zu erhalten. Außerdem sollen zum eigentlichen Zweck des Webauftritts unnötige Verbindungsaufbauten blockiert und mit einer Vergleichsmessung der Datenmengen vor und nach dem Blockieren die Ressourcenschonung und damit die Verbesserung der Nachhaltigkeit darstellt werden. Abschließend werden Möglichkeiten zum Selbstschutz bzw. Selbstverteidigung angegeben.
Dieses Projekt ist außerdem durch die gegebene Aufgabenstellung (Anhang E.1) vom AMSL/ITI motiviert.

2 Stand der Technik

Als Ausgangspunkt für unsere Arbeit haben wir uns an [1] orientiert, wo bereits ein Untersuchungskonzept mit entsprechenden Reporting Tools für Apps beschrieben wurde. Allerdings beziehen wir zusätzlich zum statischen Untersuchungsansatz dynamische Elemente in die Analyse mit ein; beispielsweise untersuchen wir das Verhalten einer Webseite nach Bestätigen des Cookie-Banners.

2.1 Werkzeugauswahl

Der *Testerstick* [2] - oder SPASS-Stick - wird als eine virtuelle Maschine auf Debian-Basis verwendet, die eine datensparsame und einheitliche Testumgebung für unsere Analysen darstellt. Zusätzlich kommt er mit allen im nachfolgenden genannten Tools vorinstalliert und spart deshalb signifikant Zeit bei der Einrichtung und Vorbereitung.

2.1.1 Reporting Tools

PrivacyScore [3] ist ein online Webseiten-Analysetool, welches wir hauptsächlich für die sicherheitsrelevanten Informationen verwenden, wie z.B. Schwachstellen (CSP, XFO).
webbkoll [4] ist ein weiteres Webseiten-Analysetool, welches sich mit *PrivacyScore* ergänzt und für dieselbe Gründe verwendet wird.
Der EU-EDPS *Website Evidence Collector* [5] ist ein auf JavaScript basiertes Tool zur Analyse von Webseiten. Dieses nutzt *ungoogled Chromium* [6] als Browser ohne aktivierte Schutzmaßnahmen, damit potentiell gefährliche Seiten trotzdem geladen werden. Das GitHub-Repository [7] von Tim Reiprich stellt eine Implementierung eines vollautomatischen Ansatzes zum Akzeptieren von Cookie-Bannern für den *Website Evidence Collector* zur Verfügung. Dabei wird versucht, aus dem Quelltext der zu untersuchenden Webseite die HTML-DOM-Struktur des Cookie-Banner-Accept-Buttons mit Hilfe einer Keyword-Suche zu identifizieren. Die Verlässlichkeit dieses Ansatzes soll in dieser Arbeit evaluiert und ggf. mit alternativen Möglichkeiten verglichen werden. Zusammen mit der Erweiterung der Cookie-Banner-Umgehung nutzen wir den *Website Evidence Collector* hauptsächlich zum Finden von Trackern und Cookies.

2.1.2 Webbrowser und Addons

Der *Firefox ESR* [8] wird verwendet, um den Datenverkehr einer Webseite beim aufrufen bzw. benutzen nachzuvollziehen und die Effektivität unserer Blockiermaßnahmen einordnen zu können.
NoScript [9] ist ein Browser-Addon, welches Skripte von Webseiten blockieren kann. Es arbeitet nach dem Whitelist-Prinzip und wird von uns verwendet um Skripte, die nicht

zur Funktionalität der Webseite beitragen, zu blockieren und somit Daten zu sparen.
uBlockOrigin [10] ist ein weiteres Addon für den Browser, es wird von uns für denselben Zweck wie *NoScript* verwendet, funktioniert jedoch nach dem Blacklist-Prinzip. Die beiden Tools können sich so ergänzen.

2.1.3 Analysetool: Wireshark

Wireshark [11] ist ein Netzwerkanalysetool, welches wir für eine ausführlichere Analyse des Datenverkehrs der untersuchten Webseiten sowie zum Aufbrechen der TLS-Verschlüsselung [12] verwenden. Unter anderem bietet das Tool eine Aufschlüsselung der Daten nach Zeit und Informationen zur Gesamtanzahl der gesendeten und empfangene Pakete.

3 Konzept

3.1 Konzepte zur Umgehung von Cookie-Bannern mit dem Website Evidence Collector [Bernhard]

Die Werkzeuge *PrivacyScore* und *webbkoll* bieten die Möglichkeit, Webseiten anhand deren URL auf sicherheits- sowie datenschutzrelevante Aspekte zu überprüfen. Dabei kommt es häufig vor, dass zu untersuchende Webseiten einige Inhalte oder Drittanbieterdienste erst dann einbinden bzw. laden, wenn das Cookie-Banner akzeptiert wurde. Es kann deshalb passieren, dass das Verhalten einer Webseite bei einer Untersuchung mit den o.g. Tools von dem eines realen Aufrufs mit einem Webbrowser deutlich abweicht. Im Rahmen dieser Arbeit sollen zwei mögliche Ansätze zur Umgehung bzw. zum Akzeptieren von Cookie-Bannern am Beispiel des *Website Evidence Collector*s vergleichend analysiert werden, inwiefern diesem Problem zukünftig begegnet werden kann. Ziel ist es, realistischere Ergebnisse im Vergleich zu den zuvor erwähnten Analysetools *PrivacyScore* und *webbkoll* zu erhalten, da Cookie-Banner immer noch häufig akzeptiert werden [13].

3.1.1 Ansatz 1: Manuelles Extrahieren von Consent-Cookies

Der folgende Ansatz geht davon aus, dass die Entscheidung des Benutzers, den Cookie-Banner zu akzeptieren, widerum als sog. „**Consent-Cookies**“ beim Client gespeichert werden. Consent-Cookies werden oftmals encodiert gespeichert und/oder sind zeitbeschränkt gültig und werden erst nach dem Akzeptieren des Cookie-Banners übertragen. Deshalb müssen die benötigten Consent-Cookies (im Idealfall direkt) vor der eigentlichen Untersuchung von Hand mit einem Webbrowser (*Firefox ESR*) manuell von der Webseite ausgelesen werden. Die konkrete Vorgehensweise der Cookie-Extraktion sowie der Übergabe der Cookies an den *Website Evidence Collector* wird genauer in Abs. 4.1.1 beschrieben. Liegen dem *Website Evidence Collector* die aktuellen Consent-Cookies vor, entsteht bei der zu untersuchenden Webseite der Eindruck, das Cookie-Banner wäre bereits bei einem früheren Besuch des Nutzers bestätigt worden. Für die zu untersuchende Webseite bedeutet das, dass alle optionalen Drittanbieterinhalte vom internen Browser des *Website Evidence Collectors* (*ungoogled Chromium*) nachgeladen werden.

3.1.2 Ansatz 2: Automatisches Akzeptieren von Cookie-Bannern

Ein automatisiertes Akzeptieren der Cookie-Banner ist nicht trivial, weil Cookie-Banner von Webseiten auf verschiedenen und teilweise einzigartigen HTML-DOM-Strukturen basieren, was eine allgemeingültige und automatisierte Selektion des Accept-Buttons ausgehend vom Quelltext der Webseite erschwert. Trotzdem wurde mit [7] ein vollautomatischer Ansatz entwickelt, bei dem m.H. von regulären Ausdrücken eine **Keyword-Suche** durchgeführt wird, um dadurch den Accept-Button von evtl. vorhandenen Cookie-Bannern zu identifizieren und zu klicken. Der entscheidende Vorteil neben der Automatisierung besteht darin, dass alle Cookies (im Vergleich zum Ansatz aus Abs. 3.1.1) innerhalb derselben Browser-Session übertragen werden, was einem realen Besuch der Webseite am nächsten kommt. Aufgrund der Natur dieses Ansatzes kann es allerdings immer passieren, dass Webseiten Cookie-Banner-Strukturen verwenden, die nicht von der Keyword-Suche erfasst werden. Die Herausforderung besteht also darin, eine möglichst vollständige Keyword-Liste zu verwalten und ständig zu aktualisieren, da ansonsten der Ansatz wirkungslos ist. Wie der *Website Evidence Collector* konkret modifiziert werden muss, wird in Abs. 4.1.2 erläutert.

3.2 Untersuchungsmethodik [Jonas]

Um möglichst objektive und vergleichbare Ergebnisse unserer Analysen auch personen- und ortsübergreifend sicherzustellen, haben wir ein allgemeingültiges Untersuchungskonzept entwickelt. Dieses haben wir zur einfachen Reproduzierbarkeit in die Form eines Flowcharts gebracht und ist in Anhang A.1 zu finden.
Als Basis unserer Untersuchung dienen die drei Tools *PrivacyScore*, *webbkoll* und der EU-EDPS *Website Evidence Collector*. Während die ersten beiden Tools jeweils über ein Web-Interface benutzt werden können und für uns hauptsächlich zur Analyse der Sicherheitsaspekte der Webseite genutzt werden können, liefert der *Website Evidence Collector* durch die Umgehung von Cookie-Bannern die detailliertesten Ergebnisse zu Skripten und Diensten von Drittanbietern.
Für *PrivacyScore* und *webbkoll* ist es ausreichend, die Webseiten zu besuchen, die URLs der zu untersuchenden Webseite einzugeben und sich die Ergebnisse zu notieren. Für den *Website Evidence Collector*, wird das modifizierte Skript auf dem Testerstick ausgeführt, erst die Webseite angegeben, dann die nötigen Consent-Cookies zur Umgehung des Cookie-Banners, dann werden die Ergebnisse automatisch lokal gespeichert.
Nachdem alle Analysen durchgeführt wurden und die Ergebnisse gespeichert sind, ist es besonders wichtig, sämtliche gefundenen Tracker zu notieren, um die Evaluation nach abgeschlossener Analyse zu vereinfachen. Sind alle Ergebnisse notiert, können diese toolübergreifend miteinander verglichen werden.
Anschließend beginnt der manuelle Teil der Untersuchung, dieser besteht daraus, Tracker, Scripte und Cookies zu blockieren und herauszufinden, ob diese ohne Funktionseinschränkungen auf der Webseite blockiert werden können. Dafür muss zuerst eine Ausgangsbasis

gefunden werden, um spätere Ergebnisse vergleichen zu können. Dazu wird die Webseite im Testerstick mit *Firefox ESR* aufgerufen und sämtliche Blockiermaßnahmen deaktiviert, der Datenverkehr wird mit *Firefox ESR* analysiert und die Anzahl an Anfragen und die genutzte Datenmenge notiert.
Ist die Ausgangsbasis gefunden, kann mit den Blockiermaßnahmen begonnen werden. Dazu werden die Tools *uBlockOrigin* und *NoScript* verwendet. *NoScript* basiert dabei auf einem Whitelist-Prinzip und *uBlockOrigin* auf einem Blacklist-Prinzip. Damit können nun nach und nach einzelne Skripte blockiert und deren Auswirkung auf die Webseite getestet werden. Verursacht das Blockieren vom Script keine Funktionsänderungen auf der Webseite, kann es zur Liste an blockierbaren Skripten hinzugefügt werden, sonst muss es aktiv bleiben. Auf diese Art werden sämtliche Skripte analysiert. Ist dieser Teil abgeschlossen, wird die Analyse des Datenverkehrs wiederholt, diesmal jedoch mit den Blockiermaßnahmen aktiv, einmal einzeln, dann alle zusammen.
Anschließend kann mit der *Wireshark*-Analyse begonnen werden, dazu bilden wir auch hier wieder eine Ausgangsbasis ohne Blockiermaßnahmen. Da wir auch die SSL-Schlüssel für unsere Analyse benötigen, starten wir den Webbrowser über das Terminal mit dem Befehl `SSLKEYLOGFILE=/home/tester/Schreibtisch/keys.ssl firefox` und rufen die Webseite erneut auf. Auch diesen Test wiederholen wir mit den Blockiermaßnahmen aktiv.
Sind diese Tests abgeschlossen, kann der Vergleich zu den automatisierten Tools geschlossen werden. Als letzten Punkt gilt es dann, die gefundenen Informationen mit der Datenschutzerklärung der Webseite zu vergleichen, um mögliche Unterschiede festzustellen.

3.3 Evaluationsmethodik [Jonas]

Sind alle Ergebnisse gesammelt, müssen die Daten evaluiert werden. Aus den Ergebnissen von *PrivacyScore* und *webbkoll* lassen sich vor allem IT-sicherheitsrelevante Informationen ableiten, da diese dort am breitesten aufgestellt sind. Die für uns interessanten Informationen beziehen sich dabei auf die Verschlüsselung der Daten (HSTS, TLS), Mail-Verschlüsselung (`SWEET32`, Secure Client Re-Negotiation) und andere mögliche Schwachstellen der Webseite, die potentiell von Angreifern ausgenutzt werden können (XFO/CSP-Header, Referrer Policy).
Da der *Website Evidence Collector* durch unsere Änderungen jetzt in der Lage ist, Cookie-Banner zu umgehen, sind die Ergebnisse von ihm besonders im Hinblick auf Drittanbieter und Tracking interessant, da die Ergebnisse dadurch am detailliertesten sind.
Durch die Messungen mit dem *Firefox ESR* kann hauptsächlich die Effektivität unserer angewandter Blockiermaßnahmen evaluiert werden. Sind die Maßnahmen erfolgreich, sollten sich sowohl die Anfragen als auch übertragene Datenmengen in beide Richtungen deutlich reduzieren. Die Ergebnisse der *Wireshark*-Analyse können dazu verwendet werden, die Ergebnisse zu validieren und zusätzliche Daten herauszufinden, wie die Aufschlüsselung der Anfragen nach Zeit, z.B. direkt nach dem Aufrufen der Webseite oder nach dem Akzeptieren des Cookie-Banners.

4 Implementierung

4.1 Umsetzung der Umgehung von Cookie-Bannern mit dem Website Evidence Collector [Bernhard]

4.1.1 *Ansatz 1: Manuelles Extrahieren von Consent-Cookies*

Bevor mit der Consent-Cookie-Extraktion begonnen wird muss sichergestellt werden, dass keine Cookies im Browser (zumindest für diese Seite) gesetzt sind. Deshalb sollte vor den folgenden Schritten stets die Browser-History vollständig gelöscht werden. Der zuvor in Abs. 3.1.1 konzeptionierte Ansatz zur Consent-Cookie-Extraktion wird folgendermaßen implementiert:

1) **Aufruf der Webseite**:
   Die zu untersuchende Webseite wird zunächst initial im *Firefox ESR* aufgerufen. Mit `F12` können die Entwicklertools aufgerufen werden, wo unter dem Reiter „Web-Speicher“ der Punkt „Cookies“ geöffnet wird. Die dort bereits sichtbaren Cookies können keine Consent-Cookies sein, da das Cookie-Banner noch nicht akzeptiert wurde.
2) **Akzeptieren des Cookie-Banners**:
   Anschließend kann das Cookie-Banner akzeptiert werden. Dadurch werden i.d.R. neben den relevanten Consent-Cookies auch weitere Cookies im Web-Speicher gesetzt.
3) **Identifizieren der Consent-Cookies**:
   Um Consent-Cookies zuverlässig zu erkennen, gibt es keine allgemeingültige Regel. In den meisten Fällen können diese m.H. des Cookie-Namens oder durch das Ausschlussverfahren identifiziert werden. Session-Cookies können beispielsweise keine Consent-Cookies sein, da diese nur für eine Sitzung gültig sind, aber Consent-Cookies i.d.R mindestens 90 Tage gelten.
4) **Speichern der Consent-Cookies**:
   Alle Consent-Cookies müssen jeweils mit Namen und Inhalt erfasst werden. Diese Informationen sind in folgendem Format zu speichern: `cookieName=cookieContent`. Insofern mehrere Consent-Cookies identifiziert wurden, können diese mit „`;`“ getrennt werden.

Die im Rahmen dieser Arbeit extrahierten Consent-Cookies für die 6 untersuchten Webseiten sind im Anhang B.1 zu finden.
Der *Website Evidence Collector* bietet die Möglichkeit, Cookies mit einem Kommandozeilenargument an die zu untersuchende Webseite zu übergeben. Der Aufruf des EDPS wird also um folgenden Parameter ergänzt, wobei für die Cookie-Daten das in Schritt 4 beschriebene Format einzuhalten ist: `--set-cookie <cookieData>`.
Basierend darauf wurde das auf dem *Testerstick* vorhandene Wrapper-Script für den EDPS-Aufruf abgewandelt und um eine Eingabe für Cookies ergänzt (auch hier ist das in Schritt 4 beschriebene Format anzuwenden). Das erweiterte Script wurde für die in dieser Arbeit durchgeführten Untersuchungen mit dem *Website Evidence Collector* verwendet und ist in Anhang B.2 zu finden. Die somit erzielten Verbesserungen in Bezug auf die Ergebnisse der Website-Analysen werden ausführlich in Abs. 5.1 erläutert.

4.1.2 Ansatz 2: Automatisches Akzeptieren von Cookie-Bannern

Der in Abs. 3.1.2 skizzierte Ansatz wurde von Tim Reiprich im Rahmen einer vergleichenden Implementierung [7] von *PrivacyScore*, *webbkoll* und dem *Website Evidence Collector* umgesetzt. Der *Website Evidence Collector* nutzt als internen Browser einen *ungoogled Chromium*, der die Datei „`browser-session.js`“ einbindet, um die Untersuchungsfunktionalitäten zu implementieren.
Die konkrete Implementierung des vollautomatischen Ansatzes setzt an dieser Stelle an, wobei `browser-session.js` durch eine modifizierte Version [14] ausgetauscht wird, in der die Keyword-Suche sowie die Automation des Accept-Button-Klicks ergänzt wurde. Dazu wird ein neues Docker-Image erstellt, welches den erweiterten *Website Evidence Collector* kapselt. Dafür werden mehrere Dateien benötigt:

- `Dockerfile`: Das Dockerfile enthält alle benötigten Abhängigkeiten einschließlich des Patches der EDPS-Installation (Anhang C.1, Zeile 46).
- `entrypoint.sh`: Wrapper-Script für den EDPS, um die Ausführung innerhalb des Docker-Containers zu realisieren und die Ergebnisse bereitzustellen (Anhang C.2, Zeile 15).
- `build-edps.sh`: Helper-Script, um das Docker-Image für den gepatchten EDPS mit der aktuellen `browser-session.js` von [14] zu erstellen (Anhang C.3, Zeile 3 und 6).
- `run-edps.sh`: Helper-Script, um die Ausführung des Images als Container zu vereinfachen (Anhang C.4, Zeile 2).

Folgende Schritte sind zum Aufsetzen des modifizierten *Website Evidence Collector*s notwendig:

1) **Herunterladen benötigter Dateien**:
   ZIP-Archiv (1.84 KB) von [15] herunterladen und entpacken
2) **Docker-Image bauen**:
   `chmod +x ./build-edps.sh && ./build-edps.sh`
3) **Ausgabeverzeichnis erstellen und Berechtigung für EDPS setzen**:
   `mkdir ./edps-output && sudo chown 1001:1001 ./edps-output`

Anschließend können Webseiten mit folgendem Befehl untersucht werden: `./run-edps.sh <URL>`
Wie sich dieser Ansatz im Vergleich zur in Abs. 3.1.1 beschriebenen Möglichkeit verhält, wird später in Abs. 5.1 evaluiert.

4.2 Umsetzung der Website-Analysen

4.2.1 *PrivacyScore*, *webbkoll*, *Website Evidence Collector* [Jonas]

Da die Analysetools *PrivacyScore*, *webbkoll* und *Website Evidence Collector* größtenteils automatisiert sind, gestaltet sich die Umsetzung der Analysen relativ simpel. Für die Tools *PrivacyScore* und *webbkoll* müssen die Web-Interfaces aufgerufen und dann in die Suchleiste die URL der zu untersuchenden Webseite eingegeben werden. Nach einer kurzen Wartezeit, in welcher die Analyse durchgeführt wird, werden die Ergebnisse auf einer neuen Seite präsentiert.
Für unsere Analyse benötigen wir von *PrivacyScore* und *webbkoll* hauptsächlich die sicherheitsrelevanten Informationen. Diese lassen sich bei *PrivacyScore* unter den Punkten „EncWeb: Encryption of Web Traffic“, „Attacks: Protection Against Various Attacks“ und „EncMail: Encryption of Mail Traffic“ finden, bei *webbkoll* sind alle Punkte außer die Cookies und Anfragen von Drittanbietern relevant.
Die Ergebnisse der beiden Analysen lassen sich als Link speichern, werden aber von nachfolgenden Analysen derselben Webseite überschrieben, deshalb sind in unseren Ergebnistabellen die relevanten Ergebnisse von den Tools herausgeschrieben und mit Zeitstempel versehen.
Der *Website Evidence Collector* ist ein lokales Tool und muss deshalb auf der Maschine vor Benutzung eingerichtet werden, da dieser jedoch auf dem von uns verwendeten *Testerstick* bereits vorinstalliert ist, kann dieser Schritt übersprungen werden. Um die erweiterten Varianten des *Website Evidence Collector*s verwenden zu können, müssen die in Abs. 4.1 beschriebenen Schritte implementiert werden.
Um den *Website Evidence Collector* zu benutzen, müssen die vollständige URL der Webseite, die Cookie-Daten zur Umgehung des Cookie-Banners und das Verzeichnis, in welches die Ergebnisse gespeichert werden sollen, im Wrapper-Script B.2 angegeben werden, dann wird die Analyse der Webseite vollautomatisch durchgeführt und die Ergebnisse in Form einer HTML Datei gespeichert. Für den vollautomatischen Ansatz wird zum Aufruf lediglich die URL der zu untersuchenden Webseite benötigt, welche als Parameter direkt an C.4 übergeben wird.
Für uns sind bei diesen Ergebnissen hauptsächlich die Informationen zu Trackern und Cookies interessant, da diese nicht mehr durch das Cookie-Banner zurückgehalten werden können. Da die Ergebnisse lokal gespeichert werden und mit einem Zeitstempel versehen werden, ist das externe Notieren der Ergebnisse nicht zwingend notwendig und dient hier nur zur besseren Übersicht.

4.2.2 *Firefox ESR*, *NoScript*, *uBlockOrigin* [Jonas]

Die Tools *NoScript* und *uBlockOrigin* werden alle innerhalb des Webbrowsers *Firefox ESR* verwendet, der zusätzlich Netzwerkanalyse-Tools integriert. Währenddessen sind *NoScript* und *uBlockOrigin* Erweiterungen, die erst installiert werden müssen, beide sind jedoch auch im *Testerstick* vorinstalliert.
Um eindeutige und wiederholbare Ergebnisse sicherzustellen, muss darauf geachtet werden, dass sämtliche Cache-Daten des Browsers gelöscht werden, bevor eine Analyse durchgeführt wird, weil sonst bereits gesetzte Cookies oder ähnliches die Ergebnisse verfälschen können.
Die Netzwerkanalyse des *Firefox ESR* befindet sich bei den „Web Developer Tools“, zuerst führen wir dabei eine Analyse mit *NoScript* und *uBlockOrigin* deaktiviert durch, um eine Grundlage zu erhalten, anschließend werden beide Tools zunächst einzeln, dann zusammen aktiviert. Der Browser-Cache muss zwischen jedem Schritt erneut gelöscht werden.
In *NoScript* werden sukzessive neue Blockiermaßnahmen hinzugefügt und die Webseite auf Funktionalität gestestet. Gibt es keine Auswirkungen, kann das Skript blockiert bleiben. Danach wird die Netzwerkanalyse mit beiden Addons einzeln aktiv und anschließend zusammen wiederholt.

4.2.3 Wireshark [Bernhard]

Bevor mit der Analyse einer Webseite mit *Wireshark* begonnen werden kann muss sichergestellt werden, dass der Cache-Speicher des für die Untersuchung verwendeten Browsers gelöscht wurde. Dies ist notwendig, damit keine Cookies oder andere Daten im lokalen Speicher vorhanden sind, welche die Untersuchung verfälschen könnten. Außerdem ist wichtig, dass alle Browser-Addons, die evtl. Blockiermaßnahmen durchführen (wie *NoScript* oder *uBlockOrigin*) ausgeschaltet wurden, da sonst viele potentielle Verbindungen gar nicht erst aufgebaut werden.
Anschließend kann mit folgenden Schritten der Traffic einer beliebigen Webseite aufgezeichnet und analysiert werden:

1) **Starten des Webbrowsers**:
   Der Webbrowser (*Firefox ESR*) muss über das Terminal mit folgendem Befehl gestartet werden, um die verwendeten SSL-Schlüssel mitzuschreiben:
   `SSLKEYLOGFILE=/home/tester/Schreibtisch/keys.ssl firefox`
2) **Durchführen der Aufzeichnung**:
   Nachdem der Browser gestartet wurde, kann mit der *Wireshark*-Aufzeichnung begonnen werden. In *Firefox ESR* wird dazu die zu untersuchende Webseite aufgerufen. Wichtig ist nun zu warten, bis die Webseite vollständig geladen wurde (einschließlich dynamischer Inhalte). Nach einigen Sekunden kann dann noch das evtl. angezeigte Cookie-Banner akzeptiert werden, um alle Drittanbieterdienste zu erlauben. Insofern auch danach alle Inhalte vollständig geladen wurden, kann die Aufzeichnung in *Wireshark* gestoppt und der Browser geschlossen werden.
3) **Vorbereiten der post-mortem-Analyse**:
   Bevor in *Wireshark* mit der Analyse der gesammelten Daten begonnen werden kann, muss das zuvor aufgezeichnete `SSLKEYLOGFILE` (siehe Schritt 1) eingebunden werden. In folgendem Menü muss der Dateipfad gesetzt werden, um den aufgezeichneten Traffic zu entschlüsseln:
   `Bearbeiten` → `Einstellungen...` → `Protocols` → `TLS` → `(Pre)-Master-Secret log filename`
4) **Post-mortem-Analyse**:
   - **Extrahieren von einzigartigen DNS-Hosts**:
     `tshark -nr file.pcapng -Y 'dns.flags.response == 0' -T fields -e dns.qry.name | sort | uniq -c`
     Die Anzahl der Hosts kann ermittelt werden, indem die Ausgabe des Befehls mit „`wc -l`“ gepiped wird.
   - **Übertragene Pakete**:
     Die zwischen der Webseite und dem Client übertragenen Pakete können im Menü `Statistiken` → `Verbindungen` ausgelesen werden. Wahlweise kann die Betrachtungsebene zwischen Ethernet, IP oder TCP gewechselt werden.
   - **I/O Graph**:
     Um zu erkennen, zu welchem Zeitpunkt der Untersuchung wie viel Traffic erzeugt wurde, ist der I/O Graph hilfreich. Dieser kann im Menü `Statistiken` → `I/O Graph` aufgerufen werden.
   - **SSLKEYLOGFILE**:
     Die mit *Wireshark* entschlüsselten TLS-Daten können mit dem *Wireshark*-Filter „`http`“ gefiltert werden.

5 Evaluierung

5.1 Evaluierung der Umgehung von Cookie-Bannern mit dem Website Evidence Collector [Bernhard]

In dieser Arbeit wurden zwei verschiedene Ansätze zur Umgehung von Cookie-Bannern vorgestellt. Dabei funktioniert der erste Ansatz mit sog. Consent-Cookies, welche manuell extrahiert werden. Der zweite Ansatz verfolgt einen vollautomatischen Ansatz m.H. einer Keyword-Suche.
Folgend werden beide Ansätze in Bezug auf die Anzahl der gelieferten Ergebnisse, auch im Vergleich zu keiner genutzten Banner-Umgehung, evaluiert.

5.1.1 Vergleich Ansatz 1 und 2

Abbildung 1 zeigt die Anzahl der eingebundenen Third-Party-Hosts, Web Beacons (vmtl. Tracker) und First- bzw. Third-Party-Cookies aller untersuchten Webseiten.
Bei der ersten Webseite *docinsider.de* ist kein signifikanter Unterschied zu erkennen, obwohl das Banner per Ansatz 1 akzeptiert wurde, allerdings über Ansatz 2 nicht. Es gibt dazu zwei mögliche Erklärungsansätze: entweder das Cookie-Banner ist wirkungslos und fungiert nur als Dummy, oder beide Möglichkeiten zur Cookie-Banner-Umgehung sind fehlgeschlagen.
Für die Webseiten *gesundheit.de* sowie *jameda.de* waren beide Ansätze offensichtlich erfolgreich. Wie zu erwarten werden deutlich mehr Third-Party-Hosts in die Kommunikation eingebunden, wodurch widerum mehr Cookies gesetzt werden. Insgesamt ist zu erkennen, dass Ansatz 2 bei fast allen Metriken sogar minimal mehr Ergebnisse liefert. Dies ist dadurch zu erklären, dass die Untersuchung im Vergleich zu Ansatz 1 in ausschließlich einer Browser-Session stattfindet, was gewährleistet, dass die Webseite sowie Drittanbieterdienste ordnungsgemäß funktionieren.
Die Webseite *kliniken.de* setzt insgesamt sehr wenige Cookies. Trotzdem ist zu erkennen, dass Ansatz 2 im Vergleich zu Ansatz 1 nicht erfolgreich war. Ähnlich verhält es sich mit *sanego.de* und *seniorenportal.de*, obwohl beide Seiten beim Akzeptieren des Banners sehr viele Cookies setzen und Drittanbieter einbinden. Die Anzahl der Ergebnisse von Ansatz 2 stimmt bei diesen Webseiten nahezu mit den Ergebnissen ohne Umgehung des Banners überein. Dies impliziert, dass die Cookie-Accept-Buttons mit Ansatz 2 nicht identifiziert bzw. geklickt werden konnten.

5.1.2 Notwendigkeit der Cookie-Banner-Umgehung

Da Ansatz 2 bei nur 2 von 6 Webseiten die Cookie-Banner erfolgreich umgehen konnte, werden für die folgende Auswertung nur die gesammelten Daten von Ansatz 1 verwendet, um die Sinnhaftigkeit einzuschätzen. Folgend werden die durchschnittlichen Zunahmen an Ergebnissen der jeweiligen Vergleichsmetriken berechnet (ausschließlich Division durch 0):

- Third-Party-Hosts:
  $\left(\frac{15}{15} + \frac{82}{12} + \frac{43}{17} + \frac{3}{1} + \frac{175}{9} + \frac{129}{11}\right) \div 6 \approx 7.42$
- Web Beacons/Tracker:
  $\left(\frac{5}{5} + \frac{34}{4} + \frac{16}{5} + \frac{64}{4} + \frac{53}{4}\right) \div 5 = 8.39$
- First-Party-Cookies:
  $\left(\frac{3}{2} + \frac{4}{1} + \frac{6}{1} + \frac{12}{10}\right) \div 4 = 3.175$
- Third-Party-Cookies:
  $\left(\frac{5}{5} + \frac{39}{3} + \frac{21}{2} + \frac{207}{5} + \frac{138}{1}\right) \div 5 = 40.78$

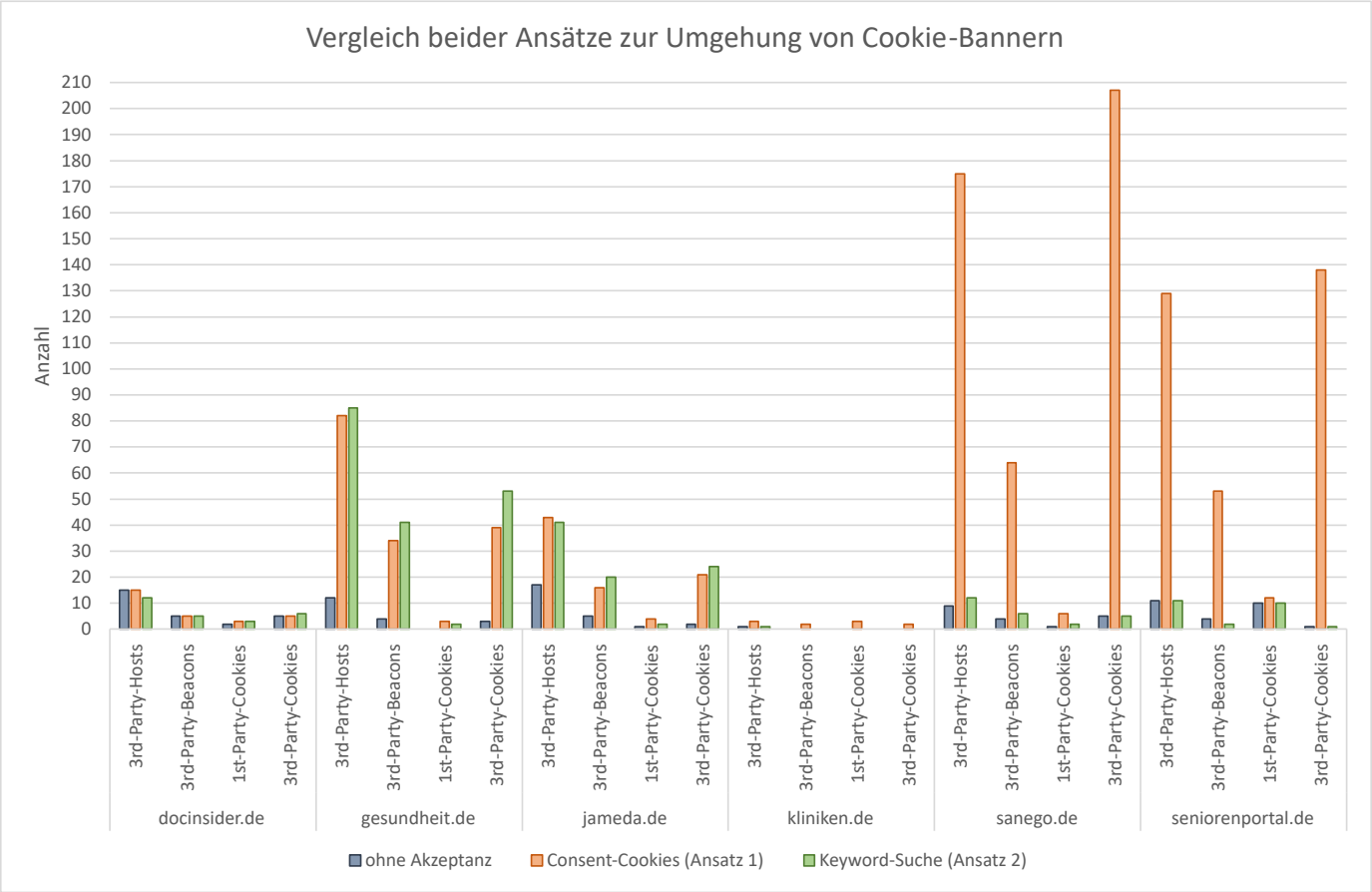


Abbildung 1: Auswirkungen der beiden Ansätze zur Umgehung von Cookie-Bannern auf die vom *Website Evidence Collector* gelieferten Ergebnisse

Da die berechneten Faktoren sehr groß sind kann festgehalten werden, dass die vorgestellten Ansätze zur Cookie-Banner-Umgehung definitiv eine Bereicherung für die Untersuchungen mit dem *Website Evidence Collector* darstellen. Insbesondere der Ansatz 1 (Consent-Cookie-Extraktion) funktioniert (zumindest für die in dieser Arbeit untersuchten Seiten) sehr zuverlässig, allerdings ist eine Skalierung auf mehr Webseiten mit sehr viel Aufwand verbunden, da das manuelle Extrahieren von Cookies vor der Analyse einen nicht zu unterschätzenden Overhead darstellt. Wenn es um eine große Anzahl von zu untersuchenden Webseiten geht, ist der vorgestellte Ansatz 2 eine besser Möglichkeit. Ansatz 2 ermöglicht eine realitätsnähere Untersuchung (Banner wird tatsächlich geklickt, alles findet in einer Browser-Session statt) und, wenn erfolgreich, i.d.R. auch minimal mehr Ergebnisse als Ansatz 1. Dazu ist es aber notwendig, die Keyword-Suche zuverlässiger zu gestalten, indem weitere, bis dato noch nicht von der Umsetzung erfasste Cookie-Banner-Strukturen analysiert und implementiert werden (siehe Abs. 6.2.1).

5.2 Auswertung der Website-Analysen

5.2.1 https://www.docinsider.de [Jonas]

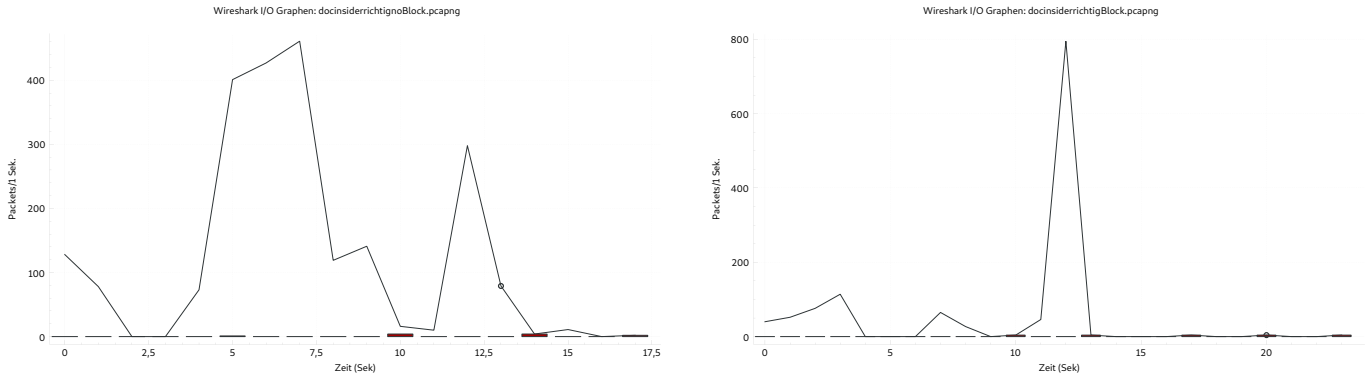
docinsider.de ist ein Arztempfehlungsportal, welches zusätzlich die Möglichkeit bietet, sich über Krankheiten zu informieren, Beschwerden zu identifizieren und Behandlungen zu finden. Die Ergebnistabelle ist im Anhang D.1 zu finden.

5.2.1.1 Drittanbieterbeteiligung

Bei *docinsider.de* liefern beide Ansätze zur Cookie-Banner-Umgehung mit dem *Website Evidence Collector* ähnliche Ergebnisse, was darauf schließen lässt, dass entweder beide Ansätze nicht funktionieren, oder das Cookie-Banner nur ein Dummy ohne Funktion ist. Diese Theorie wird auch davon bestärkt, dass die Webseite auch ohne Annahme des Cookie-Banners aufgerufen und benutzt werden kann, da dieses nur am unteren Rand angezeigt wird.
Wie auch das Diagramm 1 zeigt, ist *docinsider.de* im unteren Mittelfeld der Datensparsamkeit einzuordnen. Durch die Untersuchung mit dem *Website Evidence Collector* konnten 15 einzigartige Hosts festgestellt werden, darunter waren laut EDPS-Einordnung 9 bekannte Tracker. *docinsider.de* selbst setzt zusätzlich 2 Cookies und lässt 5 Third-Party-Cookies zu. Alle Third-Party-Cookies haben eine Laufzeit von mindestens 390 Tagen, zwei davon sogar 790 Tage.

5.2.1.2 IT-Sicherheit

docinsider.de verwendet laut *PrivacyScore* und *webbkoll* kein HSTS für Subdomains, was zur Folge hat, dass HSTS-Preloading Angriffe möglich sind. Zusätzlich konnte ein Angriffsvektor über `SWEET32` erkannt werden. Beide Tools erkannten den unzureichend implementierten CSP-Header,



(a) IO-Graph ohne Blockiermaßnahmen

(b) IO-Graph mit Blockiermaßnahmen

Abbildung 2: *Wireshark*-IO-Graph: Darstellung der Anzahl der übertragenen Pakete auf der zeitlichen Achse bei *docinsider.de*; jeweils mit bzw. ohne Blockiermaßnahmen

was das Laden von HTTP Inhalten ermöglicht und so von Angreifern ausgenutzt werden kann. Die Referrer-Policy und der XFO-Header sind auch nicht gesetzt.

5.2.1.3 Blockiermaßnahmen

Ohne Blockiermaßnahmen werden bei einem Aufruf von *docinsider.de* 81 Anfragen gestellt und 7.45 MB an Daten übertragen. Durch die geeigneten Blockiermaßnahmen können beide Werte auf die Hälfte verringert werden, ohne die Funktionalität der Webseite einzuschränken.
Das einzige Script, was bei der Blockierung mit *NoScript* die Funktionalität der Webseite einschränkt, ist *docinsider.de* selbst. Demzufolge reicht es, folgendes Skript freizuschalten:

- docinsider.de

Werden alle anderen Scripte blockiert, werden die Anfragen auf 51 und die Datenmenge auf 5.44 MB reduziert. Durch *uBlockOrigin* können die Anfragen noch weiter auf 46 reduziert werden.
Die Messungen mit *Wireshark* können diese Ergebnisse bestätigen, vor dem Block wurden 25 einzigartige Hosts angefragt, nach dem Block sind es noch 15, die Anzahl der übertragenen Pakete verringert sich im selben Maß:
Ohne die Blockierung werden 1322 Pakete vom *docinsider.de*-Server mit 2.92 MB empfangen und 661 Pakete mit 0.49 MB gesendet.
In der Abbildung 2a ist der IO-Graph dargestellt. Zu erkennen sind zwei Ausschläge, der erste entsteht direkt beim öffnen der Webseite, der zweite entsteht, sobald man auf der Seite nach unten scrollt, was zur Folge hat, dass die Werbungen vom Google Ad Manager geladen werden. Diese können aber sowohl mit *NoScript* als auch *uBlockOrigin* einfach blockiert werden.
Werden die Blockiermaßnahmen aktiviert, reduziert sich die Menge an empfangenen Paketen auf 661 (2,16 MB) und die gesendeten Pakete auf 574 (0,05 MB). Der IO-Graph der Aufzeichnung ist in Abbildung 2b dargestellt. Es lässt sich erkennen, dass diesmal nur zu einem Zeitpunkt eine Kommunikation mit der Webseite stattfindet, da keine Werbungen mehr geladen werden können. Die Menge an Paketen allgemein wurde deutlich reduziert.

5.2.1.4 Abgleich mit Datenschutzerklärung

Die Datenschutzerklärung erwähnt *Google Analytics* als Third-Party-Anbieter, jedoch wird für keine der Cookies eine Dauer der Speicherung angegeben. Wie die Daten gespeichert werden und was mit ihnen gemacht wird, wird nicht angegeben.
docinsider.de bietet alles in allem eine Webseite mit relativ geringer Anzahl an Third-Party-Diensten, da diese jedoch von Google gestellt sind, ist die Datenmenge und die Anzahl an Anfragen trotzdem verhältnismäßig groß.

5.2.2 https://www.gesundheit.de [Meryem]

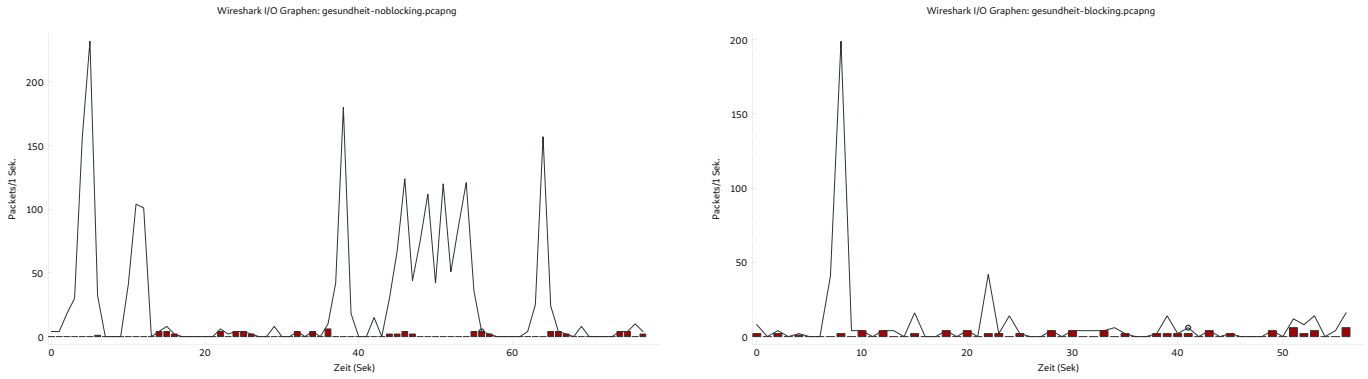
gesundheit.de bietet leicht verständliche Informationen zu allen Gesundheitsthemen für Menschen ohne medizinische Vorkenntnisse. Die Inhalte werden von Ärzten und freien Medizinautoren erstellt und regelmäßig aktualisiert. Die Ergebnistabelle ist im Anhang D.2 zu finden.

5.2.2.1 Drittanbieterbeteiligung

Für diese Webseite wurde erfolgreich der automatische Ansatz 2 zur Umgehung des Cookie-Banners verwendet. Die Webseite weist 82 Drittanbieter-Hosts auf, von denen 34 vermutlich Tracker sind. Die Anzahl der von der Webseite selbst gesetzten First-Party-Cookies beträgt 3, was im Vergleich zur Anzahl der von anderen Drittanbietern gesetzten Cookies (39) eher gering ist. Die längsten Cookies haben eine Laufzeit von 730 Tagen.

5.2.2.2 IT-Sicherheit

Basierend auf den Ergebnissen von PrivacyScore und Webbkoll ist ersichtlich, dass *gesundheit.de* kein HSTS (HTTP Strict Transport Security) implementiert hat. Diese Abwesenheit birgt ein potenzielles Risiko für die Besucher der Webseite, da HSTS für eine sichere und verschlüsselte Verbindung unerlässlich ist. Darüber hinaus ist die Sicherheit der Webseite weiterhin beeinträchtigt, da sowohl der CSP-Header als auch die Referrer-Policy nicht implementiert sind. Zusätzlich wurde die Möglichkeit eines `SWEET32`-Angriffs festgestellt.



(a) IO-Graph ohne Blockiermaßnahmen

(b) IO-Graph mit Blockiermaßnahmen

Abbildung 3: *Wireshark*-IO-Graph: Darstellung der Anzahl der übertragenen Pakete auf der zeitlichen Achse bei *gesundheit.de*; jeweils mit bzw. ohne Blockiermaßnahmen

5.2.2.3 Blockiermaßnahmen

Ohne Blockiermaßnahmen werden 436 Anfragen mit 2.98 MB an Datenübertragen. Dabei fällt auf, dass *gesundheit.de* nur auf einen Host für die Kernfunktionalität zugreifen muss. In *NoScript* wird also ausschließlich folgender Host erlaubt:

- gesundheit.de

Mit Blockiermaßnahmen reduziert sich die Anzahl der Anfragen erheblich auf nur 41 Anfragen mit 221.34 KB an übertragenen Daten, was eine signifikante Verbesserung darstellt. Die Verwendung einer einzelnen Blockiermaßnahme ist ebenfalls wirksam. *uBlockOrigin* konnte die Anfragen auf 55 reduzieren, mit 492.80 KB an übertragenen Daten, was etwas weniger effektiv ist als *NoScript* (43 Anfragen, 392.15 KB an übertragenen Daten).
Durch die Verwendung von *Wireshark* zur weiteren Untersuchung lässt sich feststellen, dass ohne Blockiermaßnahmen 3144 Pakete (3502 KB) an den Client gesendet wurden, während 2910 Pakete (747 KB) ins Internet übertragen wurden (siehe Abb. 3a).
Mit Blockiermaßnahmen wurden nur 1574 Pakete (2345 KB) an den Client gesendet und 1388 Pakete (162 KB) ins Internet übertragen. Der zugehörige IO-Graph ist in Abbildung 3b dargestellt.

5.2.2.4 Abgleich mit Datenschutzerklärung

Die Datenschutzrichtlinie von *gesundheit.de* enthält eine Opt-out-Option für Webanalyse, die es den Besuchern ermöglicht zu verhindern, dass ihre Besuche von dem Matomo-Webanalyse-Tool verfolgt werden. Durch Aktivierung der Opt-out-Option können Benutzer sicherstellen, dass ihre Aktivitäten auf der Webseite nicht aufgezeichnet werden. Die Cookies, die auf dieser Webseite verwendet werden, wurden ebenfalls in der Datenschutzerklärung aufgeführt. Zusammen mit ihrem Ablaufdatum und ihren Domainnamen stimmen sie mit den Cookies überein, die wir in unserer Untersuchung gefunden haben.

5.2.3 http://www.jameda.de [Bernhard]

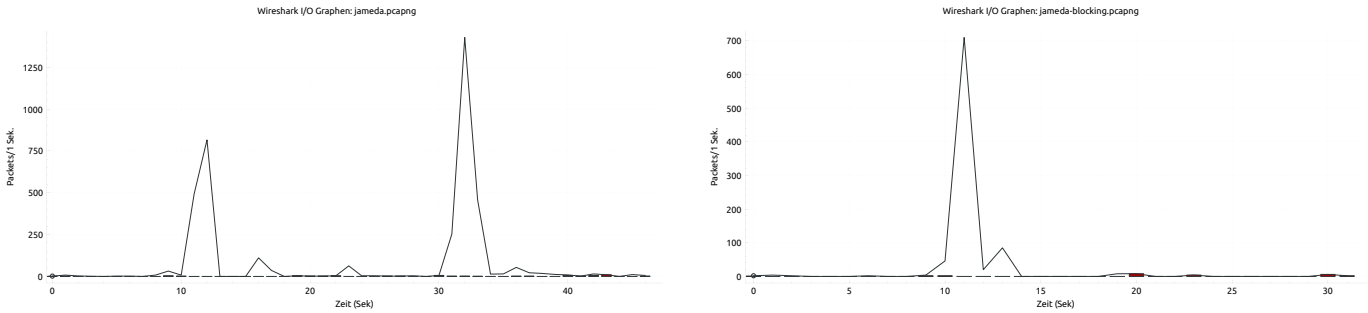
jameda.de ist eine Webseite, welche die Möglichkeit einer Arztsuche (in der Umgebung) sowie Terminvereinbarungen online anbietet. Die hier ausgeführten Ergebnisse sind in kompakter Form im Anhang D.3 zu finden.

5.2.3.1 Drittanbieterbeteiligung

Die Drittanbietereinbindung auf *jameda.de* konnte mit dem *Website Evidence Collector* und dem in Abs. 3.1.2 vorgestellten Ansatz 2 erfolgreich vollautomatisch untersucht werden. Mit 41 Third-Party-Hosts (davon 20 vermutlich Tracker) befindet sich die Webseite im oberen Mittelfeld im Vergleich mit den anderen untersuchten Webseiten. Neben großen bekannten Trackingdiensten wie *Google Analytics* oder *doubleclick* werden vermutlich auch Informationen über sog. **Tracking-Pixel** übertragen, wodurch ein schlichtes Blockieren von JavaScript-Scripten (z.B. durch *NoScript*) als Gegenmaßnahme nicht ausreicht. Während der Untersuchung wurden 14 First-Party- und 12 Third-Party-Cookies gesetzt, von denen zwei eine Gültigkeitsdauer von 400 Tagen aufweisen. Interessanterweise haben dieselben zwei Cookies, wenn die Webseite über Ansatz 1 untersucht wird, eine Gültigkeitsdauer von 730 Tagen, obwohl die Untersuchung am gleichen Tag durchgeführt wurde. Dies bestätigt, dass sich Webseiten bei Ansatz 1 der Cookie-Banner-Umgehung teilweise anders verhalten als bei Ansatz 2.

5.2.3.2 IT-Sicherheit

Von *PrivacyScore* als auch *webbkoll* wurde festgestellt, dass *jameda.de* prinzipiell HSTS aktiviert hat. Trotzdem gilt die HSTS-Implementierung nicht für Subdomains und bietet die potentielle Möglichkeit eines HSTS-Preloading-Angriffs. Weiterhin konnten mit *PrivacyScore* durch *testssl.sh* zwei potentielle Angriffsvektoren identifiziert werden (`LUCKY13` und `BREACH`), wobei die Schwere als gering bzw. mittel eingestuft wurde. Problematischer hingegen ist die Tatsache, dass *jameda.de* kein HTTP Public Key Pinning (HPKP) benutzt, was Angreifern ermöglichen kann, ungültige Zertifikate zu verwenden. Dies kann beispielsweise dazu verwendet werden, um einen Man-in-the-Middle Angriff zu realisieren. Zusätzlich implementiert *jameda.de* keinen Cross-Site-Scripting (X-XSS) Protection-Header, wodurch ein potentieller XSS-Angriff vom Browser (zumindest ohne zusätzliche Addons) nicht blockiert werden kann.



(a) IO-Graph ohne Blockiermaßnahmen

(b) IO-Graph mit Blockiermaßnahmen

Abbildung 4: *Wireshark*-IO-Graph: Darstellung der Anzahl der übertragenen Pakete auf der zeitlichen Achse bei *jameda.de*; jeweils mit bzw. ohne Blockiermaßnahmen

Das Tool *webbkoll* konnte zusätzlich Auskunft über die (unzureichend) implementierte Content-Security-Policy (CSP) geben. Diese erlaubt unter anderem das Laden von aktiven Inhalten unverschlüsselt via HTTP bzw. FTP, was theoretisch von Angreifern ausgenutzt werden könnte. Außerdem wurde keine Referrer Policy gesetzt, wodurch die letzte besuchte Webseite von Tracking-Diensten ausgelesen werden kann. Ein letztes, jedoch nicht weniger signifikantes Problem stellt die nicht eingebundene Subresource Integrity dar, wodurch insgesamt 29 Resourcen (teilweise unverschlüsselt) über HTTP und/oder ohne Prüfsumme übertragen werden.

5.2.3.3 Blockiermaßnahmen

Bei einem Besuch von *jameda.de* werden ohne Blockiermaßnahmen 121 Anfragen ausgelöst. Dabei werden 2.10 MB an Daten übertragen, was für einen einfachen Aufruf einer Webseite sehr viel ist und definitiv nicht im Sinne der Nachhaltigkeit ist.
Um Blockiermaßnahmen mit *NoScript* sinnvoll umzusetzen, benötigt *jameda.de* Zugriff auf folgende Hosts, um alle Kernfunktionalitäten bereitzustellen:

- jameda.de
- docplanner.com
- sentry.io
- maps.googleapis.com

Durch die Blockierung aller anderen Hosts wird der erzeugte Traffic auf 46 Anfragen mit nur noch 807.38 KB beschränkt. Im Vergleich ist *uBlockOrigin* zwar etwas weniger effektiv als *NoScript*, trotzdem kann man bei 55 Anfragen mit 994.57 KB von einer deutlichen Verbesserung im Sinne des Resourcenverbrauch sprechen. Werden beide Addons miteinander kombiniert, werden die besten Ergebnisse erzielt: 45 Anfragen und 806.79 KB übertragenen Daten.
Bei einer Vergleichsmessung mit *Wireshark* wurden ohne Blockiermaßnahmen 54 einzigartige DNS-Anfragen erfasst, mit *NoScript* und *uBlockOrigin* aktiviert wurden lediglich 11 einzigartige Hosts angefragt. Dies ist auch an der Anzahl von übertragenen Paketen sichtbar:
Ohne Blockiermaßnahmen wurden 2288 Pakete mit 2.923 MB Daten wurden an den Client geschickt und 1519 Pakete mit 302.46 KB ins Internet übertragen. Die zeitliche Dimension der Übertragungen ist in Abbildung 4a dargestellt. Es sind deutlich zwei Ausschläge zu erkennen: die erste Spitze an Traffic steht mit dem initialen Aufruf der Webseite in Verbindung, der zweite Peak wurde ausgelöst, nachdem das Cookie-Banner akzeptiert wurde. Daraus kann abgeleitet werden, dass das Cookie-Banner in keinem Fall akzeptiert werden sollte, da erst danach ein Großteil von Drittanbietern eingebunden wird.
Im Vergleich mit aktivierten Blockiermaßnahmen wurden deutlich weniger Pakete in beide Richtungen erfasst. Im Zeitraum der Untersuchung wurden 575 Pakete mit 995.81 KB aus dem Web empfangen und 298 Pakete mit 37.53 KB vom Client gesendet. Der zugehörige IO-Graph ist in Abbildung 4b zu sehen. Der erste und einzige Peak im Diagramm wurde durch den initialen Aufruf der Webseite ausgelöst. Durch die angewandten Blockiermaßnahmen wurde das Cookie-Banner gar nicht erst angezeigt und konnte deshalb auch nicht akzeptiert werden.

5.2.3.4 Abgleich mit Datenschutzerklärung

Die Datenschutzerklärung [16] von *jameda.de* scheint relativ vollständig, allerdings nicht in allen Punkten wirklich umgesetzt zu sein:
Direkt im Abschnitt der Begriffdefinition wird auf die Verwendung von personenbezogenen Daten zwecks Profiling und Anwendung von Tracking-Technologien wie z.B. Tracking-Pixel hingewiesen, und auch welche Daten dabei erfasst werden (können).
Weiterhin ist aus dem Abschnitt zur Übermittlung personenbezogener Daten an Drittländer zwar beschrieben, dass die von der Europäischen Kommission erlassenen EU-Standarddatenschutzklauseln Anwendung finden. Allerdings ist die Formulierung „[...] Garantien [...] hinsichtlich eines angemessenen Datenschutzniveaus im Drittland“ relativ unscharf, da unklar bleibt, was „angemessen“ in einem Drittland bedeutet. Dies ist durchaus relevant, da laut *PrivacyScore* die von *jameda.de* verwendete Server-Infrastruktur in den USA lokalisiert ist.
Weiterhin ist kritisch zu sehen, dass in den von *jameda.de* eingebundenen Tracking-Services keine Parameter für die GDPR (GDPR-Consent) zu finden sind, was ansonsten bei allen anderen untersuchten Webseiten, welche viele Drittanbieter einbinden, gegeben ist. Das bedeutet, dass die eingebundenen Dienste keine Auskunft darüber haben, ob und wie das Cookie-Banner akzeptiert wurde.

5.2.4 https://www.kliniken.de [Jonas]

kliniken.de ist eine Webseite, welche Informationen über medizinische Einrichtungen wie Krankenhäuser und Altenheime liefert, diese enthalten unter anderem Kontaktdaten und Informationen über Fachabteilungen. Die Ergebnistabelle ist im Anhang D.4 zu finden.

5.2.4.1 Drittanbieterbeteiligung

Die Untersuchung mit dem *Website Evidence Collector* konnte für *kliniken.de* nicht vollautomatisch durchgeführt werden, es wurde stattdessen die Variante aus Abs.3.1.1 zur Cookie-Banner Umgehung verwendet.
Mit nur 3 einzigartigen Hosts benutzt *kliniken.de* die geringste Menge an Drittanbietern, Cookies und Trackern. *Google Analytics* wird zwar von der Seite implementiert, ist im Cookie-Banner jedoch als optionales Cookie aufgeführt und wird standardmäßig abgelehnt, was der Datensparsamkeit sehr entgegen kommt.
Es werden 3 First-Party-Cookies und 2 Third-Party-Cookies von der Webseite eingesetzt, die Third-Party-Cookies haben dabei eine Laufzeit von 2 Jahren.

5.2.4.2 IT-Sicherheit

Auch für den Punkt der IT-Sicherheit gehört *kliniken.de* mit zu den besten untersuchten Inhalten, so können sowohl *PrivacyScore* als auch *webbkoll* nur wenige Sicherheitsrisiken feststellen. HSTS ist implementiert, was eine Verschlüsselung der Kommunikation gewährleistet, jedoch nicht für Subdomains. Beide Tools bemängeln zusätzlich die fehlende Referrer-Policy, die erweitertes Tracking ermöglicht. Ein CSP-Header wird auch nicht gesetzt, wodurch potentiell XSS-Attacken möglich sind.

5.2.4.3 Blockiermaßnahmen

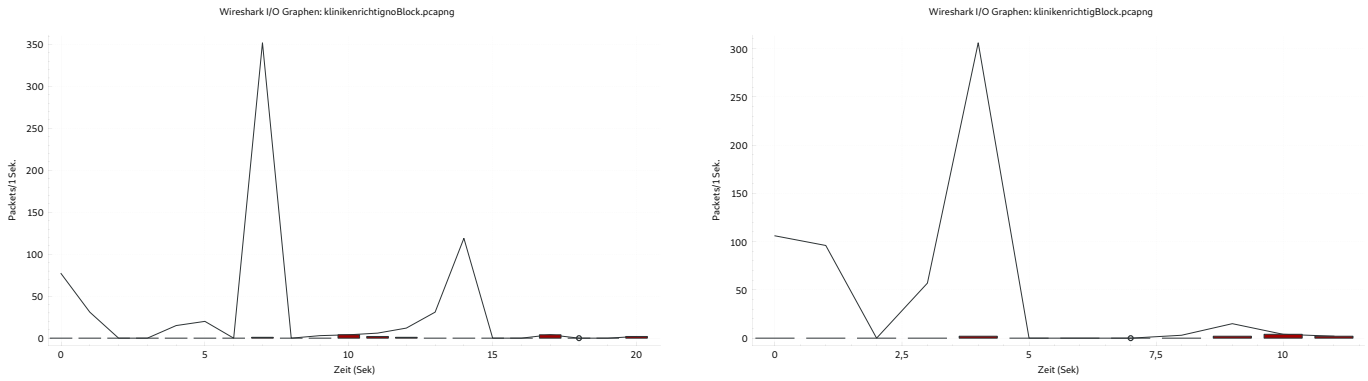
Bei einem Besuch von *kliniken.de* ohne Blockiermaßnahmen werden 18 Anfragen mit 1.44 MB Daten übertragen, für die geringe Anzahl an Anfragen und Drittanbietern, ist diese Menge an Daten trotzdem verhältnismäßig groß. Ohne Funktionalitätsverlust kann mit *NoScript* kann genau ein Skript nicht blockiert werden:

- kliniken.de

Da dieses Skript aber auch das einzige ist, was *NoScript* erkennt, kann durch dieses Tool auf der Webseite keine weitere Verbesserung erreicht werden. Auch der Dienst *uBlockOrigin* stößt an ähnliche Limitationen und kann keine Anfragen oder Datenverbrauch verhindern, wenn es eingesetzt wird.
Diese Beobachtungen werden von den *Wireshark*-Messungen bestätigt. Beide Messungen vor und nach Blockierung enthalten genau gleich viele Datenpakete und Menge an Daten darin. Durch Veränderung der Internetbandbreite und nicht exakt gleiche Länge der Tests, haben die beiden IO-Graphen trotzdem kleine Unterschiede. In beiden Fällen werden bei einem Besuch von *kliniken.de* 354 Pakete mit 0.65 MB empfangen und 322 Pakete mit 0.04 MB gesendet. In Abbildung 5a ist der IO-Graph der Aufzeichnung vor dem Block zu sehen. Der erste Ausschlag wird direkt mit dem Aufrufen der Webseite ausgelöst, der zweite kleinere Ausschlag entsteht nach dem Akzeptieren des Cookie-Banners. Beide Ausschläge sind aber im Vergleich mit anderen untersuchten Webseiten gering. Die Messung mit beiden Addons aktiv, lässt keine Änderung feststellen, wird der Vollständigkeit halber trotzdem unter Abbildung 5b eingebunden. Dieselben Ausschläge wie vorher sind auch hier zu erkennen, auch die Menge an versendeten Daten ist gleich.

5.2.4.4 Abgleich mit Datenschutzerklärung

Die Datenschutzerklärung scheint vollständig zu sein und alle wichtigen Punkte zu erwähnen, ob die Daten von Nutzern tatsächlich gut geschützt und anonymisiert werden, lässt sich an dieser Stelle nicht überprüfen.
Vorbildlich ist die komplette Auflistung aller benötigten und optionalen Cookies; die Cookie-Dauer in der Datenschutzerklärung stimmt auch mit den Ergebnissen unserer Untersuchung überein. Sämtliche personenbezogene Daten bleiben laut Datenschutzerklärung im europäischen Ausland, abgesehen von den Daten, die über *Google Analytics* erhoben werden. Da dieses Tool aber standardmäßig deaktiviert ist, ist es kein Problem für den durchschnittlichen Nutzer. Die Server von *kliniken.de* befinden sich innerhalb der EU.



(a) IO-Graph ohne Blockiermaßnahmen

(b) IO-Graph mit Blockiermaßnahmen

Abbildung 5: *Wireshark*-IO-Graph: Darstellung der Anzahl der übertragenen Pakete auf der zeitlichen Achse bei *kliniken.de*; jeweils mit bzw. ohne Blockiermaßnahmen

5.2.5 https://www.sanego.de [Bernhard]

sanego.de ist ein Dienst der die Möglichkeit bietet, Ärzte und Medikamente zu bewerten, Nebenwirkungen zu dokumentieren und Auskunft über diverse Krankheiten und Gesundheitsfragen zu erhalten. Die zugehörige Ergebnistabelle ist im Anhang D.5 zu finden.

5.2.5.1 Drittanbieterbeteiligung

Bei *sanego.de* musste auf Ansatz 1 (Abs. 3.1.1) zur Cookie-Banner-Umgehung mit dem *Website Evidence Collector* zurückgegriffen werden, da der vollautomatische Ansatz das Cookie-Banner nicht erfolgreich akzeptieren konnte. Wie auch bereits aus dem Diagramm in Abbildung 1 entnommen werden kann, ist *sanego.de* die Webseite, die von allen untersuchten Seiten am wenigsten datensparsam ist. Während der Untersuchung mit dem *Website Evidence Collector* wurden 175 einzigartige Hosts angefragt, unter denen sich 64 vermeintliche Tracking-Dienste befinden. Dabei wurden auch wieder mehrere potentielle Tracking-Pixel verwendet, um eventuelle Blockiermaßnahmen des Benutzers zu umgehen. *sanego.de* selbst setzt zwar nur 6 First-Party-Cookies, durch die hohe Anzahl an eingebundenen Drittanbieterdiensten liegt die Anzahl von Third-Party-Cookies allerdings bei 207, was von allen untersuchten Webseiten das Maximum ist. Rund die Hälfte der Drittanbieter-Cookies haben eine Laufzeit von 365 Tagen oder länger, wobei die längste Laufzeit ein Cookie von *Amazon Adsystem* mit 1838 Tagen (gut 5 Jahre) hat.

5.2.5.2 IT-Sicherheit

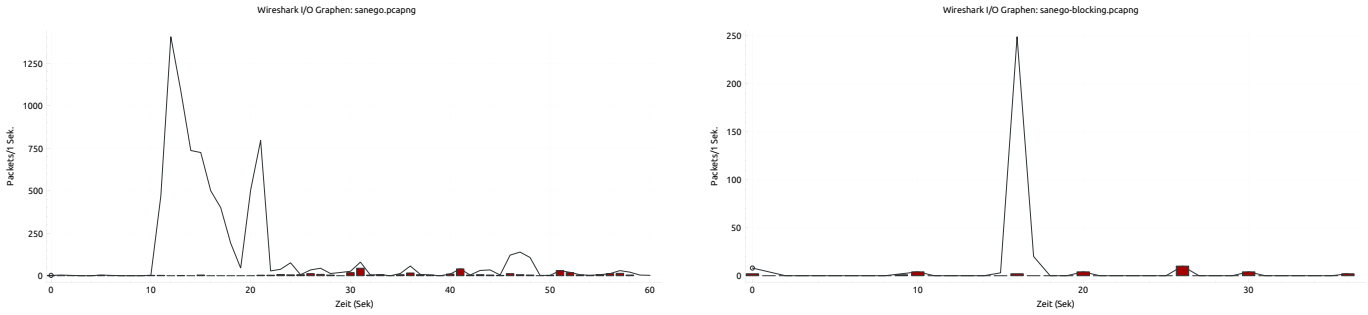
sanego.de verwendet laut *PrivacyScore* und *webbkoll* auf der Hauptseite HSTS, wodurch gewährleistet ist, dass bei der Kommunikation eine Verschlüsselung angewandt wird. Dies gilt allerdings nicht für mögliche Subdomains und leider bietet *sanego.de* auch keine HSTS-Preloading-List an. Beide Tools weisen außerdem auf die fehlende Referrer-Policy hin, wodurch erweitertes Tracking ermöglicht wird. Außerdem fehlt bei *sanego.de* auch der X-XSS-Protection-Header sowie der X-Frame-Option-Header, wodurch schädlicher Code m.H. von Werbung auf der Webseite platziert werden kann.
Über *testssl.sh* (von *PrivacyScore*) konnten weiterhin folgende potentielle Schwachstellen identifiziert werden: `BREACH`, `BEAST`, `LUCKY13`. Zusätzlich ist *sanego.de* anfällig für die Downgrade-Attacke „`TLS_FALLBACK_SCSV`“ (RFC 7507). Weiterhin konnte festgestellt werden, dass kein HPKP genutzt wird, wodurch ein Angriffsvektor für MITM-Angriffe existiert.
Das Tool *webbkoll* weißt zusätzlich auf die unzureichend implementierte Content-Security-Policy (CSP) hin, wodurch unter anderem **Click-Jacking** möglich ist. Die Subresource-Integrity wurde ebenfalls vernachlässigt, da 8 Skripte genutzt werden, welche ohne zusätzliche Sicherheitsprüfung (beispielsweise mit einer Prüfsumme) von Third-Party-Hosts eingebunden werden.

5.2.5.3 Blockiermaßnahmen

Ohne Blockiermaßnahmen werden bei einem Aufruf von *sanego.de* 365 Anfragen ausgelöst, wodurch 2.82 MB Daten übertragen werden. Da dies deutlich mehr ist als notwendig, sollte durch geeignete Blockiermaßnahmen der Resourcenverbrauch geschont werden.
Für eine funktionalitätserhaltende Konfiguration von *NoScript* reicht es, *sanego.de* selbst als einzigen Host zu erlauben:

- sanego.de

Da alle anderen Hosts blockiert sind, werden lediglich nur noch 14 Anfragen mit 468.57 KB bei einem Besuch von *sanego.de* initiiert. Im Vergleich dazu schafft es *uBlockOrigin* sogar, den Datenaustausch auf 11 Anfragen mit 468.49 KB zu begrenzen. Diese Werte können selbst mit einer Kombination beider Addons nicht weiter unterboten werden.
Die mit *Wireshark* durchgeführte Vergleichsmessung bestätigt die oben genannten Werte: ohne Blockiermaßnahmen wurden 134 einzigartige DNS-Hosts angefragt, mit Blockiermaßnahmen wurden lediglich 9 einzigartige DNS-Hosts benötigt, um die Webseite zu laden. Dementsprechend konnte auch die Anzahl der übertragenen Pakete deutlich reduziert werden:
Ohne Blockiermaßnahmen wurden während des Besuchs von *sanego.de* 4143 Pakete mit 3.78 MB vom Client empfangen und 151 Pakete mit insgesamt 486.586 KB gesendet. In Abbildung 6a ist der IO-Graph der Aufzeichnung zu sehen. Der erste sehr große Ausschlag wurde direkt nach dem initialen Aufruf der Webseite ausgelöst. Nach dem Akzeptieren des Cookie-Banners wurden weitere Daten übertragen, allerdings deutlich



(a) IO-Graph ohne Blockiermaßnahmen

(b) IO-Graph mit Blockiermaßnahmen

Abbildung 6: *Wireshark*-IO-Graph: Darstellung der Anzahl der übertragenen Pakete auf der zeitlichen Achse bei *sanego.de*; jeweils mit bzw. ohne Blockiermaßnahmen

weniger als beim initialen Aufruf. Daraus kann geschlossen werden, dass *sanego.de* bereits viele Drittanbieterdienste bzw. Tracker einbindet, ohne auf die Entscheidung des Benutzers zu warten, was gerade für die Datensparsamkeit eine Katastrope dargestellt. Die einzige Möglichkeit dies zu verhindern besteht darin, *NoScript* und/oder *uBlockOrigin* zu verwenden, um Drittanbieter-Anfragen im Vorraus zu unterbinden.
Mit aktivierten Blockiermaßnahmen konnte die Effizienz während des Aufrufs der Webseite teilweise deutlich verbessert werden. Die Anzahl von empfangenen Paketen ist mit 3380 immer noch sehr hoch, allerdings wurden dabei nur 700.706 KB an Daten gesendet. Vom Client wurden nach den umgesetzten Blockiermaßnahmen nur noch 140 Pakete mit 15.578 KB ins Internet übertragen. Der IO-Graph der Aufzeichnung ist in Abbildung 6b dargestellt. Zu erkennen ist, dass nur zu einem Zeitpunkt eine Kommunikation mit der Webseite stattfand. Ein Cookie-Banner wurde nicht angezeigt und konnte deshalb auch nicht akzeptiert werden. Während des Peaks wurden nur ein 5-tel der Pakete im Vergleich zu den Datenmengen ohne Blockierungen beim Aufruf der Webseite übertragen, was eine deutliche Verbesserung darstellt.

5.2.5.4 Abgleich mit Datenschutzerklärung

In der Datenschutzerklärung [17] von *sanego.de* werden folgende Informationen genannt, die m.H. von Cookies erhoben wurden: Browser, Betriebssystem, IP-Adressse, Zeitpunkt, Referrer, aufgerufene Daten/Links. Dabei wird mehrfach erwähnt, dass die Daten von *sanego.de* anonymisiert und nur für 30 Tage gespeichert werden. Dies ist etwas irreführend, da rund die Hälfte der bei der Untersuchung erfassten Third-Party-Cookies eine Laufzeit von über 365 Tagen, einzelne sogar bis zu 1838 Tagen aufweisen. Ob die erhobenen Daten wirklich von *sanego.de* intern anonymisiert bzw. pseudonymisiert werden, kann an dieser Stelle nicht geklärt werden.
sanego.de bindet von allen untersuchten Webseiten am meisten Drittanbieterdienste ein und setzt dementsprechend auch die meisten Third-Party-Cookies. Trotzdem ist in allen aufgezeichneten Anfragen der GDPR-Consent-Token vorhanden, wodurch es (zumindest theoretisch) möglich ist, dass Tracking-Anbieter die im Cookie-Banner eingestellten Optionen auch umsetzen können. Der Webserver von *sanego.de* ist laut *PrivacyScore* in Deutschland lokalisiert.

5.2.6 https://www.seniorenportal.de [Meryem]

seniorenportal.de ist eine Webseite, die nützliche Informationen und Tipps als Anlaufstelle für die erfahrene Generation und ihre Angehörigen bereitstellt. Es verfügt auch über einen Spielebereich und einen Online-Shop mit einer vielfältigen Produktauswahl. Die zugehörige Ergebnistabelle ist im Anhang D.6 zu finden.

5.2.6.1 Drittanbieterbeteiligung

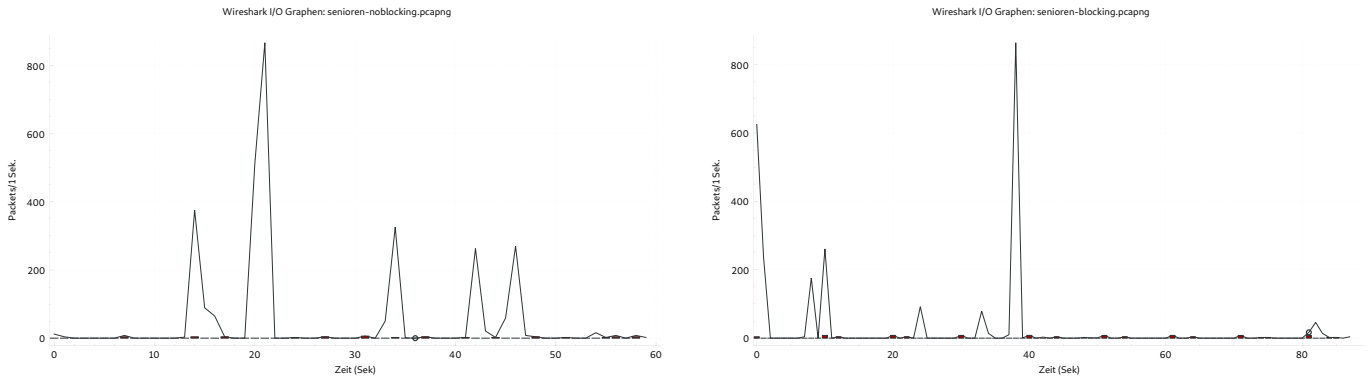
Bei *seniorenportal.de* war der Ansatz 2, um die Cookie-Banner zu umgehen, nicht erfolgreich, wie in Abbildung 1 gezeigt wird. Das Akzeptieren des Cookie-Banners (mit dem zweiten Ansatz) oder das Ablehnen führt zu keiner wirklichen Änderung, weshalb in diesem Fall der erste Ansatz verwendet wurde. Die Untersuchung mit dem *Website Evidence Collector* ergab folgende Ergebnisse: 129 Drittanbieter-Hosts mit 53 potentiell möglichen Trackern, 12 First-Party-Cookies und 138 Drittanbieter-Cookies mit einer Haltbarkeit von bis zu 1866 Tagen.

5.2.6.2 IT-Sicherheit

Laut *webbkoll* und *PrivacyScore* implementiert *seniorenportal.de* kein HSTS und weist weitere wichtige Sicherheitsmaßnahmen nicht auf. HSTS ist eine kritische Sicherheitsfunktion, die Webbrowsern anweist, nur sichere, verschlüsselte HTTPS-Verbindungen zu verwenden, wenn sie auf die Webseite zugreifen. Zusätzlich sind der XFO-Header und der CSP-Header nicht gesetzt, was die Webseite anfällig für Sicherheitsrisiken wie Click-Jacking und Cross-Site-Scripting-Angriffe macht. Ebenso sind der X-Content-Type-Header und die Referrer-Richtlinie nicht gesetzt, was potenzielle Schwachstellen darstellt.

5.2.6.3 Blockiermaßnahmen

Ohne jegliche Blockierungsmaßnahmen wurden 130 Anfragen generiert, was zu einem Datentransfer von



(a) IO-Graph ohne Blockiermaßnahmen

(b) IO-Graph mit Blockiermaßnahmen

Abbildung 7: *Wireshark*-IO-Graph: Darstellung der Anzahl der übertragenen Pakete auf der zeitlichen Achse bei *seniorenportal.de*; jeweils mit bzw. ohne Blockiermaßnahmen

1.21 MB führte, als die Webseite besucht wurde. Es ist erwähnenswert, dass laut *NoScript* die Webseite für ihre Kernfunktionalität nur Zugriff auf einen Host benötigt:

- seniorenportal.de

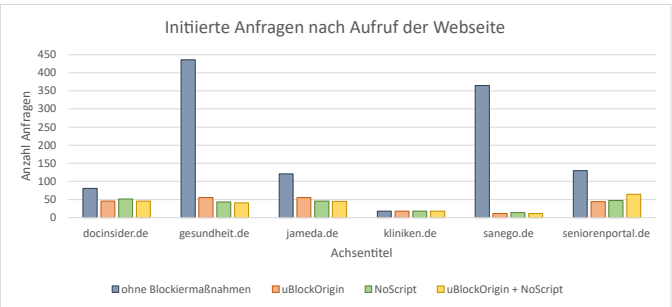
Überraschenderweise ist die Verwendung einer einzelnen Blockierungsmaßnahme effektiver als die Verwendung beider Blockierungsmaßnahmen zusammen. Wenn *NoScript* als Blockierungsmaßnahme verwendet wurde, reduzierte sich die Anzahl der Anfragen auf 47, wobei 247.72 KB Daten übertragen wurden. Andererseits wurden bei Verwendung von *uBlockOrigin* als alleinige Blockierungsmaßnahme 44 Anfragen mit 274.26 KB Datenübertragung durchgeführt, was *uBlockOrigin* zur effektivsten Maßnahme macht. Interessanterweise waren beide Addons zusammen weniger effektiv als bei Verwendung nur einer Blockierungsmaßnahme. In diesem Fall wurden 64 Anfragen mit einem Datentransfer von 452,12 Kilobyte durchgeführt.
In der *Wireshark*-Aufzeichnung über einen bestimmten Zeitraum wurden ohne jegliche Blockierungsmaßnahmen 116 eindeutige Anfragen gestellt, und 3647 Pakete (2153 KB) wurden an den Client gesendet, wobei 2273 Pakete (1978 KB) ins Internet übertragen wurden. Der entsprechende IO-Graph (7)a veranschaulicht diese Ergebnisse visuell. Als Blockierungsmaßnahmen aktiv waren, wurden nur 42 DNS-Anfragen gestellt, und 1880 Pakete (1999 Kilobyte) wurden an den Client übertragen, während 745 Pakete (203 Kilobyte) ins Internet übertragen wurden (siehe IO-Graph in Abb. 7b).

5.2.6.4 Abgleich mit Datenschutzerklärung

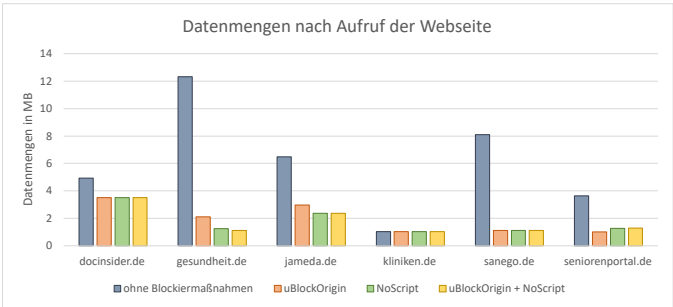
Die Webseite bietet Benutzern die Möglichkeit, sich auf der Grundlage berechtigter Interessen durch Eingabe ihrer persönlichen Daten zu registrieren. Während der Registrierung werden Daten wie IP-Adresse, Datum, Uhrzeit und Browsereinstellungen erfasst und gespeichert. Es erfolgt keine Weitergabe von Daten an Dritte. Die Einwilligung der Benutzer wird während des Registrierungsprozesses eingeholt. Benutzer haben die Möglichkeit, die Analyse mithilfe von Cookies abzulehnen. Darüber hinaus werden die IP-Adressen der Benutzer teilweise anonymisiert. Die Daten werden für 180 Tage gespeichert und *Google Analytics* wird für die Webanalyse verwendet, wobei die Daten an Google Server in den USA übertragen werden. *Google Analytics* anonymisiert die IP-Adressen, um personenbezogene Daten zu schützen. Natürlich können wir nicht wissen, ob dies tatsächlich der Fall ist. Zudem stimmen die aufgelisteten Tracker mit denen überein, die in der Untersuchung gefunden wurden.

5.3 Evaluierung der Notwendigkeit von Blockiermaßnahmen [Meryem]

uBlockOrigin arbeitet nach dem Prinzip der Verwendung von Filterlisten, die spezifische Regeln enthalten, um unerwünschte Elemente, Domains oder Ressourcen zu blockieren. Dadurch implementiert es einen Blacklist-Ansatz. Im Gegensatz dazu ist *NoScript* der Whitelist-Ansatz, da es Inhalte nur von autorisierten Quellen zulässt.
Aus Gründen der Nachhaltigkeit ist es notwendig, mindestens ein Addon zu verwenden. Dabei ist *uBlockOrigin* die empfohlene Wahl (für den Fall, dass nur ein Addon verwendet wird), da es Tracking-Pixel effektiv blockiert, vergleichbare Ergebnisse wie *NoScript* liefert (siehe Abb. 8) und für den Otto-Normalverbraucher wesentlich einfacher zu bedienen ist (als *NoScript*). Für optimale Ergebnisse sollten allerdings stets beide Addons verwendet werden.
Der Vergleich zeigt deutlich, dass der Einsatz von Blockiermaßnahmen die Anzahl der Anfragen und die Menge der übertragenen Daten signifikant reduziert, insbesondere wenn beide Addons verwendet werden. Dies verbessert nicht nur die Leistung bzw. Ladezeit der Webseite, sondern schützt auch die Privatsphäre der Benutzer, indem die Übertragung von Daten an Drittanbieter minimiert wird.



(a) Ausgelöste Drittanbieter-Anfragen mit bzw. ohne Blockiermaßnahmen



(b) Übertragene (unkomprimierte) Datenmengen mit bzw. ohne Blockiermaßnahmen

Abbildung 8: Vergleich des Einflusses von Blockiermaßnahmen auf die eingebundenen Drittanbieter und den Datenverbrauch

6 Zusammenfassung und Ausblick

6.1 Zusammenfassung der Untersuchungsergebnisse

Die von uns untersuchten Webseiten unterscheiden sich teilweise sehr stark voneinander in Bezug auf die Einbindung von Drittanbieterdiensten. Neben Extrembeispielen wie *sanego.de* mit über 100 verschiedenen eingebundenen 3rd-Party-Hosts gibt es auch Webseiten wie *kliniken.de*, die wenige bis keine unnötigen Dienste einbinden. Allerdings wurde vom *Website Evidence Collector* bei 4 von 6 untersuchten Seiten eine zweistellige Anzahl von vermeintlichen Trackern identifiziert, d.h. diese Webseiten sind nicht datensparsam.

Durch die übermäßige Nutzung von Drittanbieterdiensten sind die Datenmengen bei einem Aufruf dieser Webseiten teilweise viel höher als notwendig. Deshalb sollte für die Nachhaltigkeit im Sinne des Resourcenverbrauchs und als Möglichkeit der Selbstverteidigung stets mindestens ein Blocking-Ansatz verfolgt werden, Whitelist-Blocking (*NoScript*) oder Blacklist-Blocking (*uBlockOrigin*). Die Wirksamkeit beider Ansätze ist dabei hoch und relativ ähnlich, wobei *NoScript* in den meisten Fällen noch etwas mehr blockiert als *uBlockOrigin*. Optimale Ergebnisse werden erreicht, wenn beide Addons miteinander kombiniert verwendet werden.

Die vorgestellten Möglichkeiten zur Umgehung der Cookie-Banner haben beide ihre Vor- und Nachteile. Für eine Untersuchung von einigen wenigen Webseiten ist die **Consent-Cookie-Extraktion** der bevorzugte Ansatz, da dieser sehr zuverlässig funktioniert. Das langfristige Ziel sollte allerdings sein, Cookie-Banner vollautomatisch m.H. einer **Keyword-Suche** zu akzeptieren. Dadurch erfolgt die Datenakquise wie bei einem realen Besuch der Webseite in einer einzigen Browser-Session, was i.d.R. zu mehr und besseren Ergebnissen führt.

6.2 Ausblick für zukünftige Arbeiten

6.2.1 Verbesserung der Keyword-Suche

Wie in Abb. 1 sichtbar, funktioniert der vollautomatische Ansatz zur Cookie-Banner-Akzeptanz noch nicht bei allen Webseiten. Dies liegt entweder daran, dass die Keyword-Suche kein Ergebnis liefert, oder dass das von der Keyword-Suche ermittelte Element selbst keinen onClick-Listener implementiert. In einer weiterführenden Arbeit könnte eine größere Menge von Webseiten auf ihre HTML-DOM-Struktur untersucht werden, um die Suche zu verbessern.

6.2.2 Dynamische Analyse mit besseren Tools vertiefen

Die Untersuchung in dieser Arbeit verfolgt in erster Linie einen statischen Untersuchungsansatz mit einigen wenigen dynamischen Elementen (z.B. Akzeptieren des Cookie-Banners). Der dynamische Aspekt könnte in einer weiterführenden Arbeit weiter vertieft werden, indem beispielsweise Unterseiten oder Registrierungs- bzw. Anmeldevorgänge mit in die Untersuchung mit einbezogen werden. Dazu könnte beispielsweise die im GitLab-Repo von Tim Reiprich [7] bereitgestellte Implementierung verwendet werden, welche die drei genutzten Reporting-Tools in einem einzelnen Analysetool vereinigt. Dadurch wird eine vergleichende Analyse der Ergebnisse in Zukunft einfacher und schneller möglich sein, wodurch einzelne Seiten tiefgründiger betrachtet werden können.

Literatur

[1] S. Kiltz, R. Altschaffel, T. Lucke, and J. Dittmann, "Introduction to being a privacy detective: Investigating and comparing potential privacy violations in mobile apps using forensic methods," Nov. 2020. [Online]. Available: https://www.thinkmind.org/articles/securware_2020_2_80_30032.pdf

[2] "Testerstick mit statischer analyseumgebung," May 2023.

[3] "Privacyscore." [Online]. Available: https://privacyscore.org/

[4] "webbkoll." [Online]. Available: https://webbkoll.dataskydd.net/de

[5] E. D. P. Supervisor, "Edps inspection software." [Online]. Available: https://edps.europa.eu/edps-inspection-software_en

[6] "Ungoogled chromium." [Online]. Available: https://github.com/ungoogled-software/ungoogled-chromium

[7] T. Reiprich, "ovgu / gitlab," Jun. 2023. [Online]. Available: https://gitlab.informatik.uni-halle.de/ajpqa/ovgu/

[8] "Firefox esr." [Online]. Available: https://www.mozilla.org/en-US/firefox/enterprise/

[9] "Noscript." [Online]. Available: https://noscript.net/

[10] "ublockorigin." [Online]. Available: https://ublockorigin.com/

[11] "Wireshark." [Online]. Available: https://www.wireshark.org/

[12] "Wireshark tls." [Online]. Available: https://wiki.wireshark.org/TLS#tls-decryption

[13] R. Mueller, "76% ignore cookie banners – the user behavior after 30 days of gdpr," Jul. 2018. [Online]. Available: https://www.advance-metrics.com/en/blog/76-ignore-cookie-banners-the-user-behavior-after-30-days-of-gdpr/

[14] T. Reiprich, "browser-session.js," Jun. 2023. [Online]. Available: https://gitlab.informatik.uni-halle.de/ajpqa/ovgu/-/blob/main/docker/website-evidence-collector/browser-session.js

[15] T. Reiprich and B. Birnbaum, "patched edps (dockerized)," Jun. 2023. [Online]. Available: https://github.com/birnbaum01/amsl-it-security-projects/tree/main/SITSEC1/edps-patched

[16] "Datenschutzerklärung jameda.de." [Online]. Available: https://www.jameda.de/datenschutz

[17] "Datenschutzerklärung sanego.de." [Online]. Available: https://www.sanego.de/Datenschutz

ANHANG A

A.1 Ablaufdiagramm der Untersuchungsmethodik

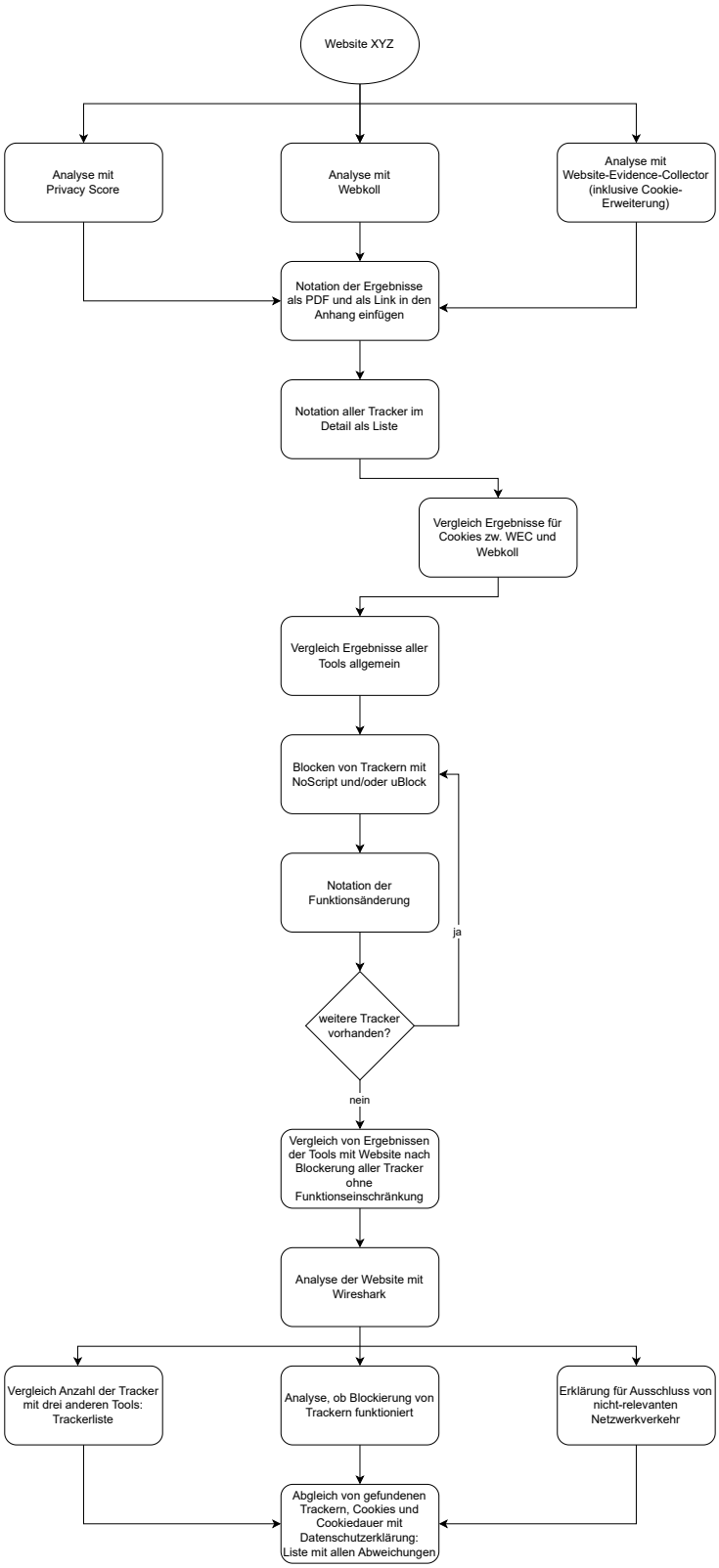


Abbildung A.1: Ablaufdiagramm der Untersuchungsmethodik

ANHANG B
COOKIE-BANNER-UMGEHUNG (ANSATZ 1): MANUELLES EXTRAHIEREN VON CONSENT-COOKIES

B.1 Extrahierte Consent-Cookies zum automatischen Akzeptieren von Cookie-Bannern m.H. des EDPS

Listing 1: Cookies für docinsider.de

```
cookies=1
```

Listing 2: Cookies für gesundheit.de

```
__cmpcccx24566=aBPtr2PxgAANgABAAOAAsAB0AFwAaAA4AB4AEUAKAApABjAEAAQQAmgB8AEOAKUBDADiQHlAPR
AigBYECyoFmALhAZEBHuCYUAcIHgUkwpKhZbC8UGJYe_A;__cmpconsentx24566=CPtqB7APtqB7AAfI2BDEDJCs
AP_AAH_AAAYgI5tb_TrfbXHC-X59fvs0OYwX1tTfA-QCABSBJ2ABwAOQ8LwGkmAaNASghiACIQwgo1ZBAAJMDEkEC
UEB4AAEAAGkAQAEhAAIIAJAgBEBQEIYAAoCAIAAAACIgAAZkAQAm1BYA-bGTGAghIAwYEgUoAgBgIIBAgIAEAAAAA
AAAAAEAAAAAAIAAIAAAAAAAQAAgjm1v9Ot9tccL5fn1--zQ5jBfW1N8D5AIAFIEnYAHAA5DwvAaSYBo0BKCGIAIhD
CCjVkEAAkwMSQQJQQHgAAQAAaQBAASEAAggAkCAEQFAQhgACgIAgAAAAIiAABmQBACbUFgD5sZMYCCEgDBgSBSgCA
GAggECAgAQAAAAAAAAAAQAAAAAAgAAgAAAAAABAACAUCgAgAyCQAQAZBoAIAMhEAEAGQqACADIZABABkOgAgAyIQA
QAZEoAIAMikAEAGQA
```

Listing 3: Cookies für jameda.de

```
OptanonAlertBoxClosed=2023-06-20T11:16:07.799Z;OptanonConsent=isGpcEnabled=0&datestamp=Tu
e+Jun+20+2023+13%3A16%3A07+GMT%2B0200+(Mitteleurop%C3%A4ische+Sommerzeit)&version=202211.
1.0&isIABGlobal=false&consentId=02a5762f-f508-48d4-90fc-06ae971cfe57&interactionCount=1&l
andingPath=NotLandingPage&groups=C0001%3A1%2CC0003%3A1%2CC0004%3A1%2CC0002%3A1&hosts=H10%
3A1%2CH77%3A1%2CH112%3A1%2CH28%3A1%2CH81%3A1%2CH82%3A1%2CH84%3A1%2CH85%3A1%2CH86%3A1%2CH7
5%3A1%2CH87%3A1%2CH11%3A1%2CH38%3A1%2CH12%3A1%2CH89%3A1%2CH92%3A1%2CH93%3A1%2CH94%3A1%2CH
32%3A1%2CH96%3A1%2CH34%3A1%2CH8%3A1&genVendors=
```

Listing 4: Cookies für kliniken.de

```
analytics-cookies-allowed=on;cookie-warning=2024-06-20T13:18:29+02:00;preferences-cookies
-allowed=on
```

Listing 5: Cookies für sanego.de

```
consentUUID=d0ebada5-33a4-47c0-a87d-4517b19b6a90_20
```

Listing 6: Cookies für seniorenportal.de

```
consentUUID=f8ddccc1-faca-4e1b-a1f8-7a2e02385452_20;euconsent-v2=CPtqTgAPtqTgAAGABCENDECs
AP_AAAAAAAYgINAZ5D5cTWFBeXx7QPs0eYwf11AVImAiChKAA6ABSDIAcLQEkmASMAyAAAACAAwEIBIBAAAkCAEEA
EAQQIAAABHkAgAEhAAIICJEABERQAAACAIKCAAAAQAIAAARIgEAmQCAQ0LmRFQAgIAQZAAAgIgAAAAEAgMAAAAAAA
IAAAAAgAAAAQAAAJBIEwACwAKgAZAA5AB4AIAAZAA0AB5AEQARQAmABPADeAHMAPwAhABDQCIAIkARwAlgBNAClAF
uAMOAfgB-gEDAI4ASYAlIBigDcAHEASIAo8BSIC8wGSAMuAawEAEAAkAD-AOcAs4CPQErALqAZCGgEABcAEMAPwAg
oBJgC0AJEAUiGAAgHUEQBQBDAD8AJMAkQBSIgACACQdA0AAWABUADIAHIAPgBAADIAGgAPAAfQBEAEUAJgAT4AuAC
6AGIAMwAbwA5gB-AENAIgAiQBLACaAFKALEAW4AwwBowD8AP0AgYBFoCOAI6ASYAlIBaADFAG4AOIAc4A6gB9gEXg
JEATIAo8BeYC-gGSAMsAZcA1UBrAEGhwBIAC4AJAA0AB_AEcAM0Ac4A7gCCgEIALOAYEA14CPQErAJiAXUAyElAbA
AWABkADgAHwAeABEACYAFwAMQAZgBDQCIAIkARwApQBbgD8AI4AWgAxQBuADqAIvASIAo8BeYDLAGsEgBIAFwBcgD
NAHcAa8A7YB9gEegJWFQBgAmABcAEcARwAtAC8xQAIAgoCPRkAQAJgAjgCOALzGAAQEekICQACwAMgBMAC4AGIAMw
AbwBHAClAFiARwAlIBaADFAHOAOoAkQBqpAAQAGgAP4AzQBzgEFAO2Aj0BMRSBKAAsACoAGQAOQAfACAAGQANAAeQ
BEAEUAJgATwApABiADMAHMAPwAhoBEAESAKUAWIAtwBowD8AP0Ai0BHAEdAJSAYoA3AB9gEXgJEAXmAvoBkgDLAGX
ANYKADgALgAkADaAH8ARwAuQBmgDnAHcAXUA14B2wEegJiAAA.YAAAAAAAAAAA
```

B.2 Wrapper-Script des Testersticks mit Ergänzung einer Cookie-Eingabe

```
#!/bin/bash
# website-evidence-collector wrapper
clear
echo
echo Willkommen zum Aufruf des Website-Evidence-Collector
echo
echo Es wird die vollstaendige URL der zu untersuchenden Webseite
echo benoetigt, d.h. mit 'http(s)' beginnend, z.B. http://www.meineseite.de
echo
echo Weiterhin wird der Verzeichnisname auf dem Schreibtisch fuer
echo die Ergebnisse benoetigt.
echo Verwenden Sie hierfuer bitte keine Sonderzeichen und Umlaute.
echo Achtung, dieses Verzeichnis wird automatisch auf dem Schreibtisch erzeugt
echo und beim Neustart der Anwendung bei Namensgeleichheit ueberschrieben"!"
echo
echo Insofern benoetigt, koennen nachfolgend Cookies angegeben werden, die beim
echo Aufruf der Website vom EDPS mit uebertragen werden sollen.
echo So koennen beispielsweise Cookie-Banner automatisch akzeptiert werden,
echo um die Untersuchung der Website in diese Richtung auszuweiten.
echo Wenn nicht benoetigt, Feld leer lassen und mit Enter bestaetigen.
echo
read -p 'Wurden alle Daten gesichert? (Ja/Nein): ' agreed
[ $agreed != 'Ja' ] && exit 1
echo
echo Alle drei Angaben werden nachfolgend abgefragt
echo
read -p 'vollstaendige URL: ' userurlvar
read -p 'Cookie-Data: ' cookiedata
read -p 'Verzeichnisname auf dem Schreibtisch: ' folder
resultpath=$HOME/Schreibtisch/$folder
echo
echo Es wurden folgende Daten entgegengenommen':'
echo $userurlvar
echo Cookie-Data: $cookiedata
echo $resultpath
echo
echo Wenn alles korrekt durchlaufen wurde, meldet sich das Skript mit einer
echo Beendigungsmeldung und der Browser wird mit den Ergebnissen gestartet.
echo
website-evidence-collector --set-cookie "$cookiedata" --overwrite --quiet --output $resultpath $userurlvar
echo
echo Die Untersuchungergebnisse werden archiviert.
echo
cp -r $resultpath /media/tester/Datenaustausch/Website-Evidence-Collector-$folder
cd /media/tester/Datenaustausch
zip -r Website-Evidence-Collector-$folder.zip Website-Evidence-Collector-$folder
rm -r /media/tester/Datenaustausch/Website-Evidence-Collector-$folder
echo
echo
echo
echo "Die Untersuchungsergebnisse wurden in das Verzeichnis"
echo $resultpath
echo "gespeichert und nach"
echo "/media/tester/Datenaustausch/"Website-Evidence-Collector-$folder.zip
echo archiviert.
sleep 10
firefox $resultpath/inspection.html
exit 1
```

ANHANG C
COOKIE-BANNER-UMGEHUNG (ANSATZ 2): AUTOMATISCHES AKZEPTIEREN VON COOKIE-BANNERN

C.1 Dockerfile für EDPS Docker-Image

```
FROM node:lts-bullseye AS dependencies
    #install additional requirements for edps
    RUN apt update

    RUN apt install libnss3-dev -y
    RUN apt install libatk1.0-0 -y
    RUN apt install libatk-bridge2.0-0 -y
    RUN apt install libcups2 -y
    RUN apt install libdrm-dev -y
    RUN apt install libxcomposite-dev -y
    RUN apt install libxdamage1 -y
    RUN apt install libxrandr-dev -y
    RUN apt install libgbm-dev -y
    RUN apt install libxkbcommon-dev -y
    RUN apt install libasound2 -y
    RUN apt install bsdmainutils -y
    RUN apt install dnsutils -y

    #optional
    RUN apt install libgconf-2-4 -y
    RUN apt install libxshmfence-dev -y

    WORKDIR /home/wec
    RUN useradd -ms /bin/bash wec
    RUN chown -R wec /home/wec

    USER wec

FROM dependencies as testssl
    RUN git clone https://github.com/drwetter/testssl.sh.git

    RUN chmod +x /home/wec/testssl.sh/testssl.sh

FROM testssl AS edps
    RUN git clone https://github.com/EU-EDPS/website-evidence-collector.git

    WORKDIR /home/wec/website-evidence-collector
    RUN npm install

    USER root
    RUN echo 'kernel.unprivileged_userns_clone=1' > /etc/sysctl.d/userns.conf
    USER wec

    WORKDIR /home/wec

    ADD browser-session.js /home/wec/website-evidence-collector/collector/browser-session.js

FROM edps AS main
    USER root
    ADD entrypoint.sh /home/wec/website-evidence-collector/entrypoint.sh
    RUN chmod +x /home/wec/website-evidence-collector/entrypoint.sh

    USER wec
    WORKDIR /home/wec/website-evidence-collector

    ENTRYPOINT ["./entrypoint.sh"]
```

C.2 Einstiegspunkt für EDPS Docker-Image

```
#!/bin/bash
internalOutput="/home/wec/edps-results"
volumeOutput="/home/wec/edps-output"

if [ "$#" -ne 1 ]; then
    echo "Error: No URL argument specified!"
    exit 255
fi
targetUrl="$@"
outId="$(echo $targetUrl | rev | cut -d "." -f 2 | cut -d "/" -f 1 | rev)-$(date +%s)"

echo "==> WEC permissions: $(id wec)"

echo "==> Executing './bin/website-evidence-collector.js --overwrite --output $internalOutput --testssl --testssl-executable /home/wec/testssl.sh/testssl.sh $targetUrl'..."
./bin/website-evidence-collector.js --overwrite --output $internalOutput --testssl --testssl-executable /home/wec/testssl.sh/testssl.sh $targetUrl
returnCode=$?

echo "==> Copying and Archiving results..."

cp -r $internalOutput $volumeOutput/edps-$outId

cd ../
tar -cvzf "$volumeOutput/edps-$outId.tar.gz" $(echo $internalOutput | rev | cut -d "/" -f 1 | rev)

if [ -f "$volumeOutput/edps-$outId.tar.gz" ]; then
    echo "==> Results saved to '$volumeOutput/edps-$outId.tar.gz'."
else
    echo "==> Failed!"
    exit 254
fi
exit $returnCode
```

C.3 Build-Script für EDPS Docker-Image

```
#!/bin/bash
if [ ! -f "./browser-session.js" ]; then
    wget -O ./browser-session.js https://gitlab.informatik.uni-halle.de/ajpqa/ovgu/-/raw/main/docker/website-evidence-collector/browser-session.js
fi
if [ -f "./browser-session.js" ]; then
    DOCKER_BUILDKIT=1 docker build --tag edps_patched .
    exit 0
else
    echo "Error: File 'browser-session.js' not found!"
    exit 1
fi
```

C.4 Ausführung von EDPS Docker-Image als Container

```
#!/bin/bash
docker run --name edps_patched --tty --rm --cap-add=SYS_ADMIN --volume=$(realpath ./edps-output):/home/wec/edps-output edps_patched:latest $@
exit 0
```

ANHANG D

D.1 Ergebnistabelle von „http://www.docinsider.de“

Tabelle 1: Übersicht über gesammelte Daten von „http://www.docinsider.de“

Werkzeug	Metrik	Ergebnisse
PrivacyScore 07.06.2023, 10:12	Drittanbieteranfragen	13 einzigartige Hosts
	Tracker	9 bekannte Tracker
	1st-Party Cookies	2 Kurzzeit-Cookies, 4 Langzeit-Cookies
	3rd-Party Cookies	1 Kurzzeit-Cookie, 1 Langezeit-Cookie von Tracker
	Verschlüsselung/HSTS	HSTS preloading Angriff möglich, TLS 1.2
	Mail-Verschlüsselung	SWEET32-Angriff möglich, Secure Client Re-Negotiation-Angriff möglich
	potentielle Schwachstellen	XFO-Header nicht gesetzt, CSP-Header nicht gesetzt, Referrer-Policy nicht gesetzt
webbkoll 07.06.2023, 10:13:37	Drittanbieteranfragen	62 Anfragen an 17 einzigartige Hosts
	Tracker	0 bekannte Tracker
	1st-Party Cookies	1 Cookie
	3rd-Party Cookies	1 Cookie
	Verschlüsselung/HSTS	TLS 1.2, HSTS implementiert, aber nicht für Subdomains
	Policies (Content-Security, Referrer)	CSP mit Fehlern implementiert
	Subresource Integrity	29 Objekte werden von Quellen ohne integrity- oder crossorigin-Attribut geladen
EDPS Website Evidence Collector 20.06.2023, 12:56:34	Drittanbieteranfragen	15 einzigartige Hosts
	Tracker	5 third-Party Web Beacons
	1st-Party Cookies	3 Cookies
	3rd-Party Cookies	5 Cookies
	Verschlüsselung/HSTS	Redirect zu HTTPS
	Social-Media-Einbindungen	Facebook
NoScript	nicht blockierbar ohne Funktionseinschränkungen	docinsider.de
	Datenmengen mit Blockierung	51 Anfragen mit 3,50 MB / 1,96 MB
uBlockOrigin	CNAME-Tracking	-
	Datenmengen mit Blockierung	46 Anfragen mit 3,50 MB / 1,96 MB
Firefox ESR	Datenmengen mit beiden/ ohne Blockierungen	ohne Blockierungen: 81 Anfragen mit 4,93 MB / 2,52 MB, mit beiden Addons: 46 Anfragen mit 3,50 MB / 1,96 MB
Wireshark	DNS-Anfragen	25 einzigartige Anfragen, 15 mit Blockiermaßnahmen
	Pakete $S \rightarrow C$	ohne Blockierungen: 1322 (2918 KB); mit Blockierungen: 661 (2155 KB)
	Pakete $C \rightarrow S$	ohne Blockierungen: 922 (129 KB); mit Blockierungen: 574 (53 KB)

D.2 Ergebnistabelle von „http://www.gesundheit.de“

Tabelle 2: Übersicht über gesammelte Daten von „http://www.gesundheit.de“

Werkzeug	Metrik	Ergebnisse
PrivacyScore 13.06.2023, 11:48	Drittanbieteranfragen	9 einzigartige Hosts
	Tracker	2 bekannte Tracker
	1st-Party Cookies	1 Kurzzeit-Cookie, 1 Langzeit-Cookie
	3rd-Party Cookies	0 Cookie
	Verschlüsselung/HSTS	HSTS nicht implementiert
	Mail-Verschlüsselung	SWEET32-Angriff möglich
	potentielle Schwachstellen	CSP-Header nicht gesetzt, Referrer-Policy nicht gesetzt
webbkoll 13.06.2023, 11:49	Drittanbieteranfragen	60 Anfrage an 9 einzigartige Hosts
	Tracker	0 Tracker
	1st-Party Cookies	2 Cookie
	3rd-Party Cookies	1 Cookie
	Verschlüsselung/HSTS	HSTS nicht implementiert
	Policies (Content-Security, Referrer)	CSP nicht implementiert, Referrers werden übermittelt
	Subresource Integrity	49 Objekte werden von Quellen ohne integrity- oder crossover-Attribut geladen
EDPS Website Evidence Collector 20.06.2023, 13:26:58	Drittanbieteranfragen	82 einzigartige Hosts
	Tracker	34 third-Party Web Beacons
	1st-Party Cookies	3 Cookies
	3rd-Party Cookies	39 Cookies
	Verschlüsselung/HSTS	Redirect zu HTTPS
	Social-Media-Einbindungen	Facebook, Twitter, Pinterest, Instagram
NoScript	nicht blockierbar ohne Funktionseinschränkungen	gesundheit.de
	Datenmengen mit Blockierung	43 Anfragen mit 1,25MB / 392,15KB
uBlockOrigin	CNAME-Tracking	-
	Datenmengen mit Blockierung	55 Anfragen mit 2,12MB / 492,80KB
Firefox ESR	Datenmengen mit beiden/ ohne Blockierungen	ohne Blockierungen: 436 Anfragen mit 12,31MB / 2,98MB, mit beiden Addons: 41 Anfragen mit 1,12MB / 221,34KB
Wireshark	DNS-Anfragen	162 einzigartige Anfragen, 22 mit Blockiermaßnahmen
	Pakete $S \rightarrow C$	ohne Blockierungen: 3144 (3502 KB); mit Blockierungen: 1574 (2345 KB)
	Pakete $C \rightarrow S$	ohne Blockierungen: 2910 (747 KB); mit Blockierungen: 1388 (162 KB)

D.3 Ergebnistabelle von „http://www.jameda.de“

Tabelle 3: Übersicht über gesammelte Daten von „http://www.jameda.de“

Werkzeug	Metrik	Ergebnisse
PrivacyScore 20.06.2023, 18:01	Drittanbieteranfragen	17 einzigartige Hosts
	Tracker	6 bekannte Tracker
	1st-Party Cookies	2 Langzeit-Cookies
	3rd-Party Cookies	1 Kurzzeit-Cookie von Tracker
	Verschlüsselung/HSTS	alle Objekte werden über HTTPS übertragen, TLS 1.2
	Mail-Verschlüsselung	SWEET32-Angriff möglich
	potentielle Schwachstellen	XFO-Header nicht gesetzt, X-XSS-Protection-Header nicht gesetzt, X-Content-Type-Header nicht gesetzt, Referrer-Policy nicht gesetzt
webbkoll 20.06.2023, 18:02:37	Drittanbieteranfragen	72 Anfragen an 17 einzigartige Hosts
	Tracker	4 bekannte Tracker (Analytics)
	1st-Party Cookies	2 Cookies
	3rd-Party Cookies	1 Cookie
	Verschlüsselung/HSTS	TLS 1.3, HSTS implementiert aber nicht für Subdomains
	Policies (Content-Security, Referrer)	CSP mit Fehlern implementiert, Referrers werden übermittelt
	Subresource Integrity	29 Objekte werden von Quellen ohne integrity- oder crossorigin-Attribut geladen
EDPS Website Evidence Collector 20.06.2023, 11:45:42	Drittanbieteranfragen	41 einzigartige Hosts
	Tracker	20 third-Party Web Beacons
	1st-Party Cookies	14 Cookies
	3rd-Party Cookies	12 Cookies
	Verschlüsselung/HSTS	Redirect zu HTTPS
	Social-Media-Einbindungen	keine
NoScript	nicht blockierbar ohne Funktionseinschränkungen	jameda.de, docplanner.com, sentry.io, maps.googleapis.com
	Datenmengen mit Blockierung	46 Anfragen mit 2.36 MB / 807.38 KB
uBlockOrigin	CNAME-Tracking	-
	Datenmengen mit Blockierung	55 Anfragen mit 2.98 MB / 994.57 KB
Firefox ESR	Datenmengen mit beiden/ ohne Blockierungen	ohne Blockierungen: 121 Anfragen mit 6.49 MB / 2.10 MB, mit beiden Addons: 45 Anfragen mit 2.36 MB / 806.79 KB
Wireshark	DNS-Anfragen	54 einzigartige Anfragen; 11 mit Blockiermaßnahmen
	Pakete $S \to C$	ohne Blockierungen: 2288 (2.923 MB); mit Blockierungen: 575 (995.813 KB)
	Pakete $C \to S$	ohne Blockierungen: 1519 (302.457 KB); mit Blockierungen: 298 (37.534 KB)

D.4 Ergebnistabelle von „http://www.kliniken.de“

Tabelle 4: Übersicht über gesammelte Daten von „http://www.kliniken.de“

Werkzeug	Metrik	Ergebnisse
PrivacyScore 15.05.2023, 13:22	Drittanbieteranfragen	0 einzigartige Hosts
	Tracker	0 bekannte Tracker
	1st-Party Cookies	0 Cookies
	3rd-Party Cookies	0 Cookies
	Verschlüsselung/HSTS	HSTS preloading Angriff möglich
	Mail-Verschlüsselung	keine Schwachstellen
	potentielle Schwachstellen	CSP-Header nicht gesetzt, Referrer-Policy nicht gesetzt
webbkoll 15.05.2023, 13:38:17	Drittanbieteranfragen	0 Anfragen
	Tracker	0 bekannte Tracker (Analytics)
	1st-Party Cookies	0 Cookies
	3rd-Party Cookies	0 Cookies
	Verschlüsselung/HSTS	HSTS implementiert aber nicht für Subdomains
	Policies (Content-Security, Referrer)	CSP nicht implementiert, Referrers werden nicht übermittelt
	Subresource Integrity	nicht eingebunden, aber alle Ressourcen laden vom gleichen Ort
EDPS Website Evidence Collector 20.06.2023, 13:30:17	Drittanbieteranfragen	3 einzigartige Hosts
	Tracker	2 third-Party Web Beacons
	1st-Party Cookies	3 Cookies
	3rd-Party Cookies	2 Cookies
	Verschlüsselung/HSTS	Redirect zu HTTPS
	Social-Media-Einbindungen	Twitter, Facebook, LinkedIn
NoScript	nicht blockierbar ohne Funktionseinschränkungen	kliniken.de
	Datenmengen mit Blockierung	18 Anfragen mit 1,04MB / 404,79KB
uBlockOrigin	CNAME-Tracking	-
	Datenmengen mit Blockierung	18 Anfragen mit 1,04MB / 404,79KB
Firefox ESR	Datenmengen mit beiden/ ohne Blockierungen	ohne Blockierungen: 18 Anfragen mit 1,04 MB / 404,79 KB, mit beiden Addons: 18 Anfragen mit 1,04 MB / 404,79 KB
Wireshark	DNS-Anfragen	15
	Pakete $S \to C$	ohne Blockierungen: 354 (645 KB); mit Blockierungen: 354 (645 KB)
	Pakete $C \to S$	ohne Blockierungen: 322 (35 KB); mit Blockierungen: 322 (35 KB)

D.5 Ergebnistabelle von „http://www.sanego.de“

Tabelle 5: Übersicht über gesammelte Daten von „http://www.sanego.de“

Werkzeug	Metrik	Ergebnisse
PrivacyScore 22.06.2023, 14:14	Drittanbieteranfragen	6 einzigartige Hosts
	Tracker	3 bekannte Tracker
	1st-Party Cookies	3 Kurzzeit-Cookies, 4 Langzeit-Cookies
	3rd-Party Cookies	1 Kurzzeit-Cookie
	Verschlüsselung/HSTS	alle Objekte werden über HTTPS übertragen
	Mail-Verschlüsselung	Secure Client Re-Negotiation-Angriff möglich
	potentielle Schwachstellen	XFO-Header nicht gesetzt, X-XSS-Protection-Header nicht gesetzt, X-Content-Type-Header nicht gesetzt, Referrer-Policy nicht gesetzt
webbkoll 22.06.2023, 14:15:16	Drittanbieteranfragen	14 Anfragen an 10 einzigartige Hosts
	Tracker	3 bekannte Tracker (Analytics)
	1st-Party Cookies	7 Cookies
	3rd-Party Cookies	0 Cookies
	Verschlüsselung/HSTS	HSTS implementiert aber nicht für Subdomains
	Policies (Content-Security, Referrer)	CSP mit Fehlern implementiert, Referrers werden immer übermittelt
	Subresource Integrity	8 Objekte werden von Quellen ohne integrity- oder crossorigin-Attribut geladen
EDPS Website Evidence Collector 20.06.2023, 13:31:19	Drittanbieteranfragen	175 einzigartige Hosts
	Tracker	64 third-Party Web Beacons
	1st-Party Cookies	6 Cookies
	3rd-Party Cookies	207 Cookies
	Verschlüsselung/HSTS	Redirect zu HTTPS
	Social-Media-Einbindungen	Instagram, Facebook, YouTube, LinkedIn
NoScript	nicht blockierbar ohne Funktionseinschränkungen	sanego.de
	Datenmengen mit Blockierung	14 Anfragen mit 1.11 MB / 468.57 KB
uBlockOrigin	CNAME-Tracking	-
	Datenmengen mit Blockierung	11 Anfragen mit 1.11 MB / 468.49 KB
Firefox ESR	Datenmengen mit beiden/ ohne Blockierungen	ohne Blockierungen: 365 Anfragen mit 8.10 MB / 2.82 MB, mit beiden Addons: 11 Anfragen mit 1.11 MB / 468.51 KB
Wireshark	DNS-Anfragen	134 einzigartige Anfragen; 9 mit Blockiermaßnahmen
	Pakete $S \rightarrow C$	ohne Blockierungen: 4143 (3.78 MB); mit Blockierungen: 151 (486.586 KB)
	Pakete $C \rightarrow S$	ohne Blockierungen: 3380 (700.706 KB); mit Blockierungen: 140 (15.578 KB)

D.6 Ergebnistabelle von „http://www.seniorenportal.de“

Tabelle 6: Übersicht über gesammelte Daten von „http://www.seniorenportal.de“

Werkzeug	Metrik	Ergebnisse
PrivacyScore 13.06.2023, 19:39	Drittanbieteranfragen	10 einzigartige Hosts
	Tracker	3 bekannte Tracker
	1st-Party Cookies	4 Kurzzeit-Cookies, 7 Langzeit-Cookies
	3rd-Party Cookies	0 Cookie
	Verschlüsselung/HSTS	HSTS nicht implementiert
	Mail-Verschlüsselung	SWEET32-Angriff möglich
	potentielle Schwachstellen	XFO-Header nicht gesetzt, CSP-Header nicht gesetzt, X-Content-Type-Header nicht gesetzt, Referrer-Policy nicht gesetzt
webbkoll 13.06.2023, 19:42	Drittanbieteranfragen	21 Anfrage an 10 einzigartige Hosts
	Tracker	2 bekannte Tracker (Analytics)
	1st-Party Cookies	11 Cookies
	3rd-Party Cookies	0 Cookie
	Verschlüsselung/HSTS	HSTS nicht implementiert
	Policies (Content-Security, Referrer)	CSP nicht implementiert, Referrers werden übermittelt
	Subresource Integrity	5 Objekte werden ohne integrity- oder crossorigin-Attribut geladen
EDPS Website Evidence Collector 20.06.2023, 13:33:42	Drittanbieteranfragen	129 einzigartige Hosts
	Tracker	53 third-Party Web Beacons
	1st-Party Cookies	12 Cookies
	3rd-Party Cookies	138 Cookies
	Verschlüsselung/HSTS	Redirect zu HTTPS
	Social-Media-Einbindungen	Facebook, Twitter
NoScript	nicht blockierbar ohne Funktionseinschränkungen	seniorenportal.de
	Datenmengen mit Blockierung	47 Anfragen mit 1,28 MB / 247,72 KB
uBlockOrigin	CNAME-Tracking	-
	Datenmengen mit Blockierung	44 Anfragen mit 1,02 MB / 274,26 KB
Firefox ESR	Datenmengen mit beiden/ ohne Blockierungen	ohne Blockierungen: 130 Anfragen mit 3,65 MB / 1,21 MB, mit beiden Addons: 64 Anfragen mit 1,30 MB / 452,12 KB
Wireshark	DNS-Anfragen	116 einzigartige Anfragen, 42 mit Blockiermaßnahmen
	Pakete $S \rightarrow C$	ohne Blockierungen: 3647 (2153 KB); mit Blockierungen: 1880 (1999 KB)
	Pakete $C \rightarrow S$	ohne Blockierungen: 2273 (1978 KB); mit Blockierungen: 745 (203 KB)

ANHANG E

E.1 Aufgabenstellung

Einleitung/Motivation:

Webauftritte im Bereich von Soziales/Gesundheit aber auch bei Bezahlsystemen sind oftmals alles andere als datensparsam. Es werden Drittparteien in die Kommunikation eingebunden, ungewollt Werbung platziert und datenschutzrelevante Nutzerdaten nicht zweckgebunden sowohl an den kontaktierten Server sowie Drittparteien übermittelt [AKL+20]. IT-Sicherheitsrelevante Schwachstellen gefährden Nutzende zusätzlich. Deshalb sollen Webauftritte mit OpenSource-Werkzeugen im Bezug auf Datensparsamkeit, Datenschutz, Nachhaltigkeit im Sinne des Ressourcenverbrauchs sowie IT-Sicherheit überprüft werden. Möglichkeiten zu Selbstschutz/Selbstverteidigung sollen erforscht werden.

Aufgaben: Idee, Ansatz und Ausgangspunkt:

Ziel ist es, auf Basis des Testersticks [Kil23f] eine Untersuchungsumgebung einzurichten, um Webauftritte aus den Sektoren Gesundheit/Soziales sowie Bezahlsysteme analysieren zu können. Die Untersuchungsmethoden des Website Evidence Collectors [Rie23] des Testersticks sollen aktualisiert und um die Fähigkeit des Überschreitens von Cookie-Bannern erweitert werden. Es sollen zum eigentlichen Zweck des Webauftritts unnötige Verbindungsaufbauten geblockt werden (Browserplugins wie noscript, ublock). Eine Vergleichsmessung der Datenmengen vor und nach dem Blockieren soll die Ressourcenschonung darlegen und damit die Verbesserung der Nachhaltigkeit. Ausgangspunkt für die Untersuchungen sind die Webauftritte:

- Meine AOK (https://meine.aok.de/),
- Webportal Kliniken (https://www.kliniken.de/),
- Deutsches Seniorenportal (https://www.seniorenportal.de/pflege),
- Klarna Sofortüberweisung (https://www.sofort.com/de/),
- Amazon Pay (https://pay.amazon.de/),

weitere geeignete Webseiten sind vom Team zu identifizieren und zu testen.

Mögliche teaminterne Aufgabenaufteilung:

1 Aktualisierung und Ergänzung Website-Evidence-Collector, 1 Teammitglied Planung und Strukturierung der Tests, 1 Teammitglied Konfiguration, Anpassung der Untersuchungsumgebung, Integration der Werkzeugbestandteile

Erwartetes Ergebnis:

Umfassende Tabellendokumente zu den Testdurchläufen und Ergebnissen vor und nach Gegenmaßnahmen, überarbeitetes/angepasstes Testerstick-Image, Dokumentation aller Änderungen am Testerstick, Report

Grundkenntnisse:

Skriptprogrammierung, Kommandozeilenprogrammierung, Auswertung großer Testdokumente, systematisches und strukturiertes Testen

Abbildung E.1: Aufgabenstellung