

# **Vergleich und Analyse des privaten Modus verschiedener Browser**

## **Computer-Forensik und Vorfallsbehandlung**

Carl Schünemann

Christoph Sell

29.08.2025

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
<b>2</b>	<b>Theoretischer Hintergrund</b>	<b>4</b>
2.1	Private Browsing . . . . .	4
2.2	Angreifermodell . . . . .	6
2.3	Private Browsing Artefakte . . . . .	8
<b>3</b>	<b>Ziel der Arbeit</b>	<b>13</b>
<b>4</b>	<b>Methodik</b>	<b>16</b>
4.1	Vorbereitung . . . . .	16
4.1.1	Browserauswahl . . . . .	16
4.1.2	Browsing Szenario . . . . .	21
4.1.3	Analysewerkzeuge . . . . .	24
4.2	Datensammlung . . . . .	26
4.3	Datenanalyse . . . . .	29
4.3.1	Common Locations . . . . .	30
4.3.2	Uncommon Locations . . . . .	34
4.3.3	Registry . . . . .	37
<b>5</b>	<b>Ergebnisse</b>	<b>41</b>
5.1	Firefox . . . . .	41
5.2	Tor . . . . .	55
5.3	Chrome . . . . .	64
5.4	Brave . . . . .	64
<b>6</b>	<b>Vergleich der Browser</b>	<b>65</b>
<b>7</b>	<b>Diskussion</b>	<b>66</b>
<b>8</b>	<b>Fazit</b>	<b>68</b>
	<b>Appendices</b>	<b>69</b>
	<b>Literaturverzeichnis</b>	<b>72</b>
	<b>Literatur</b>	<b>72</b>

# 1 Einleitung

Steigende Beliebtheit private Browsing: [10] ■ Die Verwendung von PB wurde als die beliebteste Form der Online-Privatsphäre weltweit identifiziert. ■ Aufgrund der gestiegenen Sensibilität und Öffentlichkeit für den Schutz der Privatsphäre und die Regulierung des eigenen digitalen Fußabdrucks im Internet werden PB-Technologien wahrscheinlich häufiger auf den Geräten der Nutzer eingesetzt. ■ Auch wenn es schwierig ist, endgültige Nutzungsstatistiken für solche Aktionen zu erstellen, bietet der Konsens über den Online-Datenschutz einen Einblick. Im Jahr 2016 wurde die Verwendung eines PB-Fensters als die weltweit beliebteste Form der Online-Datenschutzmaßnahme identifiziert [1]. Allein in den USA nutzen Berichten zufolge rund 33 % der Nutzer ein PB-Fenster, wobei über 70 % zugeben, ihren Internetverlauf zu löschen [2]. - Eine umfassende Studie von Montasari und Peltola (2015) ergab, dass der Erfolg des privaten Modus bei verschiedenen Browsern sehr unterschiedlich ist

Vermeintliche Privatheit beim Browsen: [19] > Verschlüsselung ■ Datenschutz und Datenverwendung sind Hauptbedenken der Internetnutzer geworden [5]. ■ Fragen wie welche Daten von Unternehmen genutzt werden, mit wem sie geteilt werden und wie wertvoll sie sind, sind heute wichtige Themen. ■ Daher versuchen Benutzer, sich so weit wie möglich zu schützen, insbesondere durch Begrenzung der Datenweitergabe. ■ Lösungen wie Verschlüsselung auf HTTP-Ebene [6] und auf DNS-Ebene [7,8] sind Standard geworden und werden den Großteil des Datenverkehrs in den nächsten Jahren abdecken. ■ Sie können jedoch nur End-to-End-Konversationen verschlüsseln, d.h. IP- und TCP- oder UDP-Informationen sind immer noch verfügbar. > VPNs ■ Eine weitere beliebte Methode zum Schutz der Privatsphäre und zur Vermeidung von Datenverwendung ist die Verwendung von Virtual Private Networks (VPNs). ■ Obwohl VPNs immer beliebter geworden sind und die meisten von ihnen den IP-Verkehr verschlüsseln und tunneln können, kann der Datenverkehr tatsächlich am Endpunkt des VPNs überwacht werden. ■ Dies bedeutet, dass Akteure zwischen dem VPN-Servernetzwerk und dem Website-Server die Daten sehen und nutzen können. ■ Der VPN-Anbieter kann sogar noch weiter gehen, da er auch die Identität des Clients kennt. > Tor und Brave: 1. Die Endpunkte der verschlüsselten Verbindungen, die von Tor und Brave hergestellt werden, nicht vollständig verschlüsselt sind. Daher können einige Informationen, wie z.B. die IP-Adresse des Benutzers, an den letzten Servern in der Kette sichtbar sein. 2. Einige Tor-Ausgangsknoten haben in der Vergangenheit die Aktivität ihrer Benutzer ausspioniert, um Daten zu sammeln und möglicherweise zu verkaufen. 3. Obwohl die Verwendung von Brave und Tor dazu beitragen kann, dass Benutzer online nicht nachverfolgt werden, werden sie nicht vor Verfolgung durch andere Methoden wie Standortverfolgung oder Geräte-Fingerprinting geschützt. 4. Schließlich können auch andere Schwachstellen in der Implementierung oder Konfiguration von Tor oder Brave dazu führen, dass Daten durchsickern und somit die Privatsphäre der Benutzer kompromittiert wird.

Immer mehr Kriminelle im Internet [12]: > Das Internet und seine Nutzer wachsen ständig, aber auch die Anzahl organisierter Verbrechen und illegale Aktivitäten nehmen zu.

“Webbrowser immer beliebter bla bla ...“ [11] > Webbrowser sind heutzutage ein wichtiger Werkzeug für Online-Aktivitäten wie Online-Banking, Online-Shopping und soziale Netzwerke.

Immer mehr Internet-Nutzer:[11] ■ Im Jahr 2019 gab es laut [13] fast 4,5 Milliarden Internetnutzer.

Zunehmende Bestrebungen nach Privatheit erschwert forensische Ermittlungen [15] > Zunehmende Verwendung von verschlüsselten Daten in der Dateispeicherung und Netzwerkkommunikation erschwert Ermittlungen. > Besonders schwierig ist das Tor-Protokoll, das sich auf den Schutz der Privatsphäre des Nutzers konzentriert. > Tor-Browser hinterlässt digitale Artefakte, die von Ermittlern genutzt werden können.

Motivation Portable Browser [7] ■ Die Beliebtheit von tragbaren Webbrowsern nimmt aufgrund ihrer bequemen und kompakten Natur sowie des Vorteils, dass Daten einfach über einen USB-Stick gespeichert und übertragen werden können, zu. ■ Entwickler arbeiten an Webbrowsern, die tragbar sind und zusätzliche Sicherheitsfunktionen wie den privaten Modus-Browsing, eingebaute Werbeblocker usw. bieten. ■ Die erhöhte Wahrscheinlichkeit, tragbare Webbrowser für schädliche Aktivitäten zu nutzen, ist das Ergebnis von Cyberkriminellen, die der Ansicht sind, dass bei der Verwendung von tragbaren Webbrowsern im privaten Modus keine digitalen Fußabdrücke hinterlassen werden. ■ Das Forschungspapier zielt darauf ab, eine vergleichende Studie von vier tragbaren Webbrowsern, nämlich Brave, TOR, Vivaldi und Maxthon, zusammen mit verschiedenen Speichererfassungstools durchzuführen, um die Menge und Qualität der aus dem Speicherauszug wiederhergestellten Daten in zwei verschiedenen Bedingungen zu verstehen, nämlich wenn die Browser-Tabs geöffnet und geschlossen waren, um forensische Ermittler zu unterstützen.

Private Browsing Motivation und Ausnutzen von Kriminellen: [14] ■ Webbrowser werden täglich genutzt, um verschiedene Online-Aktivitäten durchzuführen. ■ Webbrowser speichern eine große Menge an Daten über Benutzeraktivitäten, einschließlich besuchter URLs, Suchbegriffen und Cookies. ■ Private Browsing-Modi wurden entwickelt, um Benutzern das Surfen im Internet zu ermöglichen, ohne Spuren zu hinterlassen. ■ Dies kann von Kriminellen ausgenutzt werden, um ihre Aktivitäten zu verschleiern. ■ Experimente werden auf jeder Browser-Modus durchgeführt, um zu untersuchen, ob sie Spuren auf der Festplatte oder im Arbeitsspeicher hinterlassen.

Motivation Private Browsing mit Portablen Browsern: [18] ■ Das Internet ist ein unverzichtbares Werkzeug für alltägliche Aufgaben. ■ Neben der üblichen Nutzung wünschen sich Benutzer die Möglichkeit, das Internet auf private Weise zu durchsuchen. ■ Dies kann zu einem Problem führen, wenn private Internetsitzungen vor Computerermittlern verborgen bleiben müssen, die Beweise benötigen. ■ Der Schwerpunkt dieser Forschung liegt darauf, verbleibende Artefakte aus privaten und portablen Browsing-Sitzungen zu entdecken. ■ Diese Artefakte müssen mehr als nur Dateifragmente enthalten und ausreichend sein, um eine positive Verbindung zwischen Benutzer und Sitzung herzustellen. ■ In den letzten 20 Jahren ist das Internet für alltägliche Aufgaben, die mit stationären und mobilen Computergeräten verbunden sind, drastisch unverzichtbar geworden. ■ Benutzer wünschen sich neben der üblichen Internetnutzung auch Privatsphäre und die Möglichkeit, das Internet auf private Weise zu durchsuchen. ■ Aus diesem Grund wurden neue Funktionen für das private Browsen entwickelt, die von allen gängigen Webbrowsern unterstützt werden. ■ Unsere Forschung konzentriert sich auf die Entdeckung von Informationen von lokalen Maschinen, da die meisten Computeruntersuchungen auf der Suche und Beschlagnahme von lokalen Speichergeräten beruhen. ■ Artefakte aus privaten und portablen Browsing-Sitzungen wie Benutzernamen, elektronische Kommunikation, Browsing-Verlauf, Bilder und Videos können für einen Computerermittler signifikante Beweise enthalten. ■ Wir

werden auch flüchtige Daten analysieren, die in einer gängigen Incident-Response-Umgebung verfügbar wären.

Schwachstellen in Browsern, durch die Daten "lecken" [24] ■ Private browsing ist seit 2005 eine beliebte Datenschutzfunktion in allen gängigen Browsern. ■ Laut einer Studie (-> TODO: welche?) leiden alle Browser unter einer Vielzahl von Schwachstellen, von denen viele zuvor nicht bekannt waren. ■ Die Probleme werden hauptsächlich durch eine laxere Kontrolle von Berechtigungen, inkonsistente Implementierungen der zugrunde liegenden SQLite-Datenbank, die Vernachlässigung von Cross-Mode-Interferenzen und eine fehlende Beachtung von Timing-Angriffen verursacht. ■ Alle Angriffe wurden experimentell verifiziert und Gegenmaßnahmen vorgeschlagen.

Private Browsing Motivation und Ausnutzen von Kriminellen [21] ■ Fast alle Aspekte des Lebens nutzen bereits das Internet, um auf das Internet zugreifen zu können, wird ein Webbrowser verwendet. ■ Die Einführung des Internets hat das Leben der Menschen in vielen Bereichen verändert, darunter auch im Bereich der Kriminalität, insbesondere in der Verwendung von Webbrowser-Software für Transaktionen und Prozesse im Internet. ■ Webbrowser speichern normalerweise Informationen wie URL-Verlauf, Suchbegriffe, Passwörter und andere Nutzeraktivitäten. ■ Aus Sicherheitsgründen wurden einige Funktionen von Webbrowsern entwickelt, um den privaten Modus zu ermöglichen. ■ Leider wird diese Funktion von einigen skrupellosen Menschen für kriminelle Aktivitäten durch die Anti-Forensik genutzt, um digitale Beweise in kriminellen Fällen zu minimieren oder zu verhindern.

Auswirkung von Darknet und Tor auf Forensiker [20] ■ Personen, die Inhalte aus dem Darknet abrufen möchten, müssen nicht nur in einem regulären Browser Schlüsselwörter eingeben, sondern müssen es anonym über den TOR-Browser zugreifen, um ihre Identität wie IP-Adresse oder physische Lage zu verbergen. ■ Aufgrund dieser Tatsachen ist es für Strafverfolgungsbehörden oder digitale forensische Experten schwierig, den Ursprung des Datenverkehrs, den Standort oder die Eigentümerschaft eines Computers oder einer Person im Darknet zu lokalisieren. ■ Die Auswirkungen des Darknets traten auf, als das Federal Bureau of Investigation (FBI) im Oktober 2013 die Website Silk Road abschaltete, die ein Online-Schwarzmarkt und der erste moderne Darknet-Markt für den Verkauf illegaler Drogen war. ■ Silk Road war nur über das TOR-Netzwerk zugänglich und vom Mainstream-Web verborgen. ■ Da die meisten Darknet-Sites Transaktionen über anonyme digitale Währungen wie Bitcoin durchführen, die auf kryptografischen Prinzipien basieren, ist es für digitale forensische Experten sehr schwierig, solche Transaktionen zu verfolgen, da Benutzer und Dienste anonym sind. ■ Das Ziel dieser Arbeit besteht darin, digitale forensische Techniken zu diskutieren, um solche Darknet-Verbrechen zu behandeln.

## 2 Theoretischer Hintergrund

Einleitend werden Struktur, Motivation und die abgeleiteten Forschungsfragen diskutiert.

### 2.1 Private Browsing

Definition Web Browser: > [21] ■ Der Webbrowser ist eine Softwareanwendung zum Abrufen, Präsentieren und Durchsuchen von Informationsressourcen im Internet oder World Wide Web (WWW).  
■ Eine Informationsquelle wird durch einen Uniform Resource Identifier (URI) identifiziert und kann Webseiten, Bilder, Videos oder andere Inhalte enthalten.

> [11] ■ Ein Webbrowser ist eine Software, die es Benutzern ermöglicht, das Internet über den von ihrem Dienstanbieter bereitgestellten Zugang zu nutzen. ■ Die bekanntesten Webbrowser sind Google Chrome, Mozilla Firefox, Microsoft Edge und Brave. ■ Webbrowser werden für alltägliche Aktivitäten wie das Anschauen von Videos, das Durchsuchen von Webseiten, das Posten von Bildern oder Videos in sozialen Medien und das Herunterladen und Hochladen von Dateien genutzt. ■ Browser-Modi: Es gibt zwei verschiedene Browser-Modi: den normalen Browser-Modus und den privaten Browser-Modus.

Definition „Normal Browsing“: > [11] ■ Der normale Browser-Modus speichert alle Browser-Aktivitäten wie Caches, Cookies, Suchbegriffe, Login-Daten und URL-Verlauf auf dem Computer. ■ Die Cookies speichern Details des Benutzers wie z.B. Browsing-Muster, die anzeigen können, welche Websites der Benutzer häufig besucht oder welche Videos er/sie regelmäßig ansieht.

Definition "Private Browsing": > [22] - Deshalb wurde eine neue Funktion in die Webbrowser aufgenommen, die den Internetnutzern eine größere Kontrolle über ihre Privatsphäre ermöglicht. Diese Funktion ist als "Private Browsing" bekannt und soll es den Nutzern ermöglichen, im Internet zu surfen, ohne Datenspur auf ihrem Computer zu hinterlassen.

> [10] - Private Browsing"(PB) ist ein allgemeiner Begriff, der sich auf Mechanismen, die verhindern sollen, dass ein Nutzer Beweise für sein Web-Browsing-Verhaltens auf seinem lokalen Gerät gespeichert werden. - Von Anfang an muss betont werden, dass sich privates Surfen in diesem Zusammenhang nur auf Plattformen bezieht, die lokale Privatsphäre bieten, und dass diese von Anwendungen wie Tor (siehe <https://www.torproject.org/>) zu unterscheiden sind, die sich ebenfalls auf die Online-Privatsphäre konzentrieren, sowie von Einrichtungen, die die Verfolgung und Überwachung aus der Ferne verhindern, wie z. B. der Tracking Preference Expression des W3C (auch bekannt als "Do Not Track"). - Je nach Browser des Nutzers wird eine zugehörige PB-Funktion mit unterschiedlichen Begriffen bezeichnet: Inkognito-Modus in Chrome, InPrivate in Edge und dem inzwischen nicht mehr unterstützten Internet Explorer sowie ein "privates Fenster" in Firefox.

Geschichte Private Browsing: > [22] - Die ADbC-Funktion "Privater Browsing-Modus" wurde erstmals 2005 mit Apple Safari 2.0 eingeführt. Drei Jahre später folgte Google Chrome 1.0 (Inkognito). Später,

im Jahr 2009, führten Microsoft Internet Explorer 8 und Mozilla Firefox 3.5 ihre Versionen von privaten Browsing-Modi ein, die als InPrivate bzw. Private Browsing bekannt sind (Dan, 2010).

> [14] ■ Private Browsing-Modi haben je nach Browser unterschiedliche Namen, z.B. Incognito-Modus in Chrome, InPrivate Browsing in Internet Explorer, "Private Browsing" in Firefox und Safari. ■ erstmals 2005 von Apple Safari eingeführt, gefolgt von Google Chrome und Microsoft in 2008 und Mozilla in 2009.

Grund des privaten Modus: > [11] ■ Private Browsing Mode wurde entwickelt, um die Privatsphäre und Anonymität beim Surfen im Internet zu verbessern, indem keine Spuren und Informationen von Browsing-Aktivitäten hinterlassen werden. ■ Alle neuen Caches, die während des Surfens gespeichert wurden, werden entfernt, sobald der Browser geschlossen wird. ■ Jeder Webbrowser bietet einen privaten Browser-Modus mit unterschiedlichen Bezeichnungen an, wie InPrivate Browsing für Internet Explorer und Microsoft Edge, Incognito-Modus für Google Chrome und "Private Browsing" für Mozilla Firefox. > [1] zwei wesentliche Ziele des privaten Browsing: 1. (local) Besuchte Websites sollten im privaten Modus keine Spuren auf dem Computer des Benutzers hinterlassen. Wenn ein Familienmitglied den Browserverlauf überprüft, sollte keine Evidenz von im privaten Modus besuchten Websites gefunden werden können. 2. (website) Benutzer möchten möglicherweise ihre Identität vor den Websites, die sie besuchen, verbergen, indem sie es beispielsweise für Websites schwierig machen, die Aktivitäten des Benutzers im privaten Modus mit seinen Aktivitäten im öffentlichen Modus zu verknüpfen. Dies wird als Datenschutz vor einem Webangreifer bezeichnet. > [14] ■ Private Modus Browser sollten in der Lage sein, die von besuchten Websites hinterlassenen Artefakte auf dem Computer des Benutzers zu verhindern. ■ Browser sollten es Websites unmöglich machen, herauszufinden, ob ein bestimmter Benutzer sie zuvor besucht hat, indem sie verhindern, dass Websites die Aktivitäten von Benutzern im privaten und öffentlichen Modus verknüpfen.

Stakeholder Private Browsing: > Forensiker - [12] ■ Die Entwicklung von Datenschutzfunktionen in Browsern stellt eine Herausforderung für digitale Forensiker dar, die Beweismittel sammeln möchten, um Kriminelle zu überführen. - [10] ■ Durch die Möglichkeit des privaten Browsens besteht eine erhöhte Gefahr für illegale und schädliche Online-Aktivitäten. ■ Die meisten privaten Browsing-Modi sind so konzipiert, dass sie lokal privat sind und Daten, die auf das Surfverhalten des Benutzers hinweisen, nicht auf dem Gerät gespeichert werden. ■ Diese Handlungen können die Verfügbarkeit von Beweismaterial beeinträchtigen und stellen eine Herausforderung für Untersuchungen dar. - [10] ■ Private browsing (PB) ist eine Funktion, die seit langem auf dem Radar von forensischen Praktikern steht. ■ Risiko: PB kann dazu führen, dass potenziell beweiskräftiger Inhalt nicht auf einem lokalen Gerät gespeichert wird, was zu Untersuchungsbedenken führt. ■ PB selbst hat viele legitime Anwendungen und ist nicht per se anti-forensisch, kann aber mit anti-forensischer Absicht verwendet werden. ■ Fehlende Internetinhalte stellen ein Problem für Beweissammlung ■ Private Browsing-Modi sollten die Aktivität des Nutzers vor forensischen Tools verbergen

> Kriminelle: - [12] ■ Kriminelle nutzen vermehrt private Browser, um ihre Spuren zu verwischen und ihre illegalen Handlungen zu verbergen. ■ Cyberkriminelle nutzen Private Browsing-Modi, um digitale Spuren auf dem Gerät zu verwischen und forensische Ermittler mit leeren Händen dastehen zu lassen. > Nutzerperspektive: - [10] ■ Die Verwendung von PB wurde als die beliebteste Form der Online-Privatsphäre weltweit identifiziert. ■ Aufgrund der gestiegenen Sensibilität und Öffentlichkeit für den Schutz der Privatsphäre und die Regulierung des eigenen digitalen Fußabdrucks im Internet werden PB-Technologien wahrscheinlich häufiger auf den Geräten der Nutzer eingesetzt. ■ Auch wenn es schwierig ist, endgültige Nutzungsstatistiken für solche Aktionen zu erstellen, bietet der Konsens

über den Online-Datenschutz einen Einblick. Im Jahr 2016 wurde die Verwendung eines PB-Fensters als die weltweit beliebteste Form der Online-Datenschutzmaßnahme identifiziert [1]. Allein in den USA nutzen Berichten zufolge rund 33 % der Nutzer ein PB-Fenster, wobei über 70 % zugeben, ihren Internetverlauf zu löschen [2].

- [10] ■ Die PB-Technologie wird aufgrund der gesteigerten Sensibilität und öffentlichen Aufmerksamkeit für den Schutz der Privatsphäre voraussichtlich häufiger auf Geräten verwendet. - [22] In den letzten Jahren (2010) haben jedoch viele der bekannten Webbrowser-Hersteller ihre Besorgnis über die Privatsphäre der Nutzer beim Surfen im Internet verstärkt. - Tatsächliche Gründe: [14] in [1] Experiment von Aggarwal et al.: Werbung auf Ad-Netzwerken geschaltet wurde, um verschiedene Kategorien von Websites einschließlich Erwachsenen- und Geschenk-Websites zu bewerben, um die Nutzung des privaten Modus mit der Art der besuchten Website zu korrelieren. -> Browsing-Modus auf Erwachsenen-Websites beliebter war als auf Geschenk-Websites. > Herstellerperspektive: - [14] Angeblich lt. Hersteller: o Einkaufen von Überraschungsgeschenken auf einem Familien-PC o Planung von Überraschungspartys

Stakeholder Private Browsing: > "Forensicher Ermittler [14] ■ forensischer Ermittler kann forensische Browsing-Artefakte mit forensischen Tools und Techniken wiederherstellen > "Nutzer": - [22] > Tatsächlich ergab eine Studie, dass Private Browsing auf Websites für Erwachsene beliebter ist als auf Websites für den Geschenkekauf oder für Nachrichten. Dies deutet darauf hin, dass die Anbieter von Webbrowsern den Hauptnutzen dieses Tools möglicherweise falsch einschätzen, wenn sie es als ein Tool zum Kauf von Überraschungsgeschenken beschreiben (Aggarwal, Boneh, Bursztein, und Jackson, 2010). > "Browser Entwickler [12] Die Entwickler von Browsern haben den Mangel an Benutzerdatenschutz erkannt und einen privaten Browsermodus eingeführt, der das Schreiben von Browserdaten auf die Festplatte einschränkt oder idealerweise verhindert. - [22] Einem Artikel zufolge (Belani, Jones, 2005) behaupten die Hersteller all dieser Webbrowser, dass keine der besuchten Websites, Formularfelddaten, in die Adressleiste eingegebenen Adressen, besuchten Links und Suchanfragen auf dem lokalen Computer des Nutzers gespeichert werden (Brookman, 2010).

## 2.2 Angreifermodell

Definition Local Attacker nach [1]: - Z.B. Forensischer Prüfer - hat physischen Zugriff auf den Computer des Benutzers - versucht, auf dessen privaten Browserverlauf zuzugreifen. - beispielsweise ein Familienmitglied oder ein Freund sein, der den Computer des Benutzers nutzt, um auf dessen Browserverlauf zuzugreifen. - kann darauf installierte Programme verwenden, um Informationen zu sammeln. - hat keinen Zugriff auf die Maschine des Benutzers, bevor der Benutzer das private Surfen beendet hat. Ohne diese Einschränkung ist Sicherheit gegen einen lokalen Angreifer unmöglich. (z.B: Keylogger installieren, Benutzeraktionen aufzeichnen) - Durch die Beschränkung des lokalen Angreifers auf "forensische Untersuchungen nach dem Ereignis" kann man hoffen, Sicherheit zu gewährleisten, indem der Browser persistenten Zustandsänderungen während einer privaten Surfsitzung ausreichend löscht. - Der Angreifer wartet, bis der Benutzer den privaten Browsing-Modus verlässt, und erhält dann die vollständige Kontrolle über die Maschine. Dies bedeutet, dass der Angreifer auf forensische Daten angewiesen ist. - Während der aktiven Phase kann der Angreifer nicht mit Netzwerkelementen kommunizieren, die Informationen über die Aktivitäten des Benutzers im privaten Modus enthalten. Dies bedeutet, dass die Implementierung von Browser-seitigen Datenschutzmodi untersucht wird, nicht die serverseitigen Datenschutzmodi.



- Das Ziel des Angreifers besteht darin, für eine bestimmte Menge von HTTP-Anfragen, die er wählt, festzustellen, ob der Browser eine dieser Anfragen im privaten Browsing-Modus ausgeführt hat oder nicht. Wenn der lokale Angreifer dieses Ziel nicht erreichen kann, gilt die Implementierung des privaten Browsers als sicher. -> Local Attacker weiß, wonach er sucht!
- Es wird darauf hingewiesen, dass die Definition impliziert, dass der Angreifer nicht feststellen kann, welche Websites der Benutzer besucht hat oder was der Benutzer auf einer bestimmten Website getan hat. Darüber hinaus wird auf die Eigenschaften des privaten Browsers nicht formal eingegangen, wenn der Benutzer den privaten Browsing-Modus nie verlässt.

Problem: Local Attacker muss überarbeitet werden: [14] ■ Es wurde festgestellt, dass das Konzept des lokalen Angreifers nicht ausreichend untersucht wurde und dass neue Experimente durchgeführt werden müssen, um ein besseres Verständnis für das Phänomen zu erlangen und herauszufinden, wie sich diese Funktion auf digitale forensische Untersuchungen auswirken könnte.

Definition Web Attacker nach [1] - Z.B. ISP - versucht Online-Aktivitäten des Benutzers im privaten Modus zu verfolgen und zu identifizieren, um diese mit seinen Aktivitäten im öffentlichen Modus in Verbindung zu bringen. - durch den Einsatz von Tracking-Tools oder das Sammeln von Informationen über die IP-Adresse des Benutzers oder andere Identifikationsmerkmale erfolgen. ■ Kontrolliert die von Benutzer besuchten Websites und kann Informationen über Benutzeraktivitäten sammeln (-> z.B. ISP), aber nicht über den Computer des Benutzers. ■ Webseiten können auch verschiedene Browser-Funktionen nutzen, um Browser zu identifizieren und sie über Privatsphäre-Grenzen hinweg zu verfolgen. ■ Die Electronic Frontier Foundation hat eine Website namens Panopticklick (-> TODO: In Demo zeigen?) erstellt, die zeigt, dass die meisten Browser eindeutig identifiziert werden können, was die Ziele (1) und (2) des privaten Surfers in allen Browsern unterbricht.

Anti-Forensische Grundsätze bei Browserentwicklung, um sich gegen Web-Attacker zu schützen nach [1] ■ Browser haben drei Ziele, um die Privatsphäre der Benutzer zu schützen. o Ziel 1: Ein Benutzer, der im privaten Modus surft, soll nicht mit demselben Benutzer verknüpft werden können, der im öffentlichen Modus surft. o Ziel 2: Ein Benutzer in einer privaten Sitzung soll nicht mit demselben Benutzer in einer anderen privaten Sitzung verknüpft werden können. o Ziel 3: Eine Website soll nicht erkennen können, ob der Browser im privaten Modus ist. ■ Ziele (1) und (2) sind schwierig zu erreichen, da die IP-Adresse des Browsers von Webseiten genutzt werden kann, um Benutzer über private Browsing-Grenzen hinweg zu verfolgen. ■ Das Torbutton Firefox-Erweiterung (ein Tor-Client) macht große Anstrengungen, um Ziele (1) und (2) zu erreichen, indem es die IP-Adresse des Clients über das Tor-Netzwerk versteckt und Schritte unternimmt, um das Browser-Fingerprinting zu verhindern.

Beispiel: Web Attacker Angriffe: > IP-Angriffe [19] ■ Obwohl Nutzer Verschlüsselung oder VPNs nutzen, ist ihre Privatsphäre oft ungeschützt, da mehrere Domains gleichzeitig besucht werden oder IP-Adressen von Cloud-Providern geteilt werden. ■ Eine neue Methode zur Identifizierung von Web-Browsing wird vorgestellt, die nur auf den IP-Adressen basiert, mit denen der Nutzer verbunden war, ohne DNS Reverse-Resolution durchzuführen. ■ Diese IP-Adresse-Sequenz wird in verschiedene Deep Learning Modelle eingespeist, um die tatsächlich besuchte Website zu identifizieren. ■ Untersucht wurden auch andere Faktoren wie Abhängigkeit vom DNS-Server, Genauigkeit bei Top-Domains, Datenverstärkung durch Paket-Sampling-Simulation, Auswirkungen auf Paket-Sampling und Skalierbarkeit der Methode. ■ Mit nur 10% der Pakete konnte die besuchte Website mit einer Genauigkeit und F1-Score von 94% bis 95% identifiziert werden. > ISP als „Web attacker“, um Kundenaktivität zu verfolgen [1] ■ ISP können unsere Ergebnisse nutzen, um den Datenverkehr ihrer Kunden zu identifizieren. ■ Dies ermöglicht

ISP, Daten für Marketingzwecke zu monetarisieren, sofern sie anonymisiert und mit Zustimmung der Kunden erfolgt. ■ ISP müssen jedoch darauf achten, wer Zugriff auf Netzwerkverkehrsdaten hat. ■ Das Weitergeben dieser Daten an Dritte kann zu potenziellen Datenschutzverletzungen bei Kunden führen. ■ Hauptaufgabe ist eigentlich einfach, aber es können viele Komplikationen auftreten ■ Hauptproblem ist das sogenannte "verwickelte Netz" ■ Beim Verbinden mit einer Website muss der Webbrowser eine Kaskade von Verbindungen zu anderen Websites öffnen ■ Grund dafür sind Bilder, Anzeigen, Banner, JavaScript-Bibliotheken, Social-Media-Links und vieles mehr

## 2.3 Private Browsing Artefakte

TODO: Common vs Uncommon Locations hier ansprechen

Residuale Daten > [11] ■ Überraschenderweise besteht der private Browser in Chrome und Firefox aus wenigen residuellen Daten, die jedoch Beweise für Interessen wie Suchbegriffe, E-Mail-IDs und Passwörter liefern können ■ Residuale Daten sind Daten, die von einem Gerät entfernt wurden, aber immer noch aufgespürt werden können. ■ Diese Daten können mithilfe spezieller Tools, meist in Dateiüberresten oder lokalen Ordnern, identifiziert werden. ■ Beispiele für residuale Daten sind Link-Dateien, Log-Dateien, Registry-Dateien, Prefetch-Dateien und Browser-Verlaufsdaten. ■ Digitale Forensik kann solide elektronische Beweise aus solchen Überresten und Artefakten sammeln, um sie in Gerichtsverfahren zu verwenden.

Browser Artefakte: > [11] ■ Jeder Browser hat unterschiedliche Artefakte im RAM des Geräts gespeichert ■ Im normalen Browsing-Modus werden die Browsing-History-Details des Benutzers vor und nach dem Löschen des Verlaufs im Speicher gespeichert ■ Benutzeraktivitäten und Daten beim Surfen können in normalen Browser-Modi wie Cookies, Caches, Downloads, Verlauf, anderen sensiblen Daten und temporären Dateien verfolgt und gespeichert werden, was digitalen Forensikern bei der Suche nach Beweisen hilft.

> [12] ■ Browser speichern eine Vielzahl von Nutzerdaten, die von besuchten URLs bis zu Benutzernamen und Passwörtern reichen ■ Das Wissen, dass Browser private Surfdaten preisgeben, ist schon etwas, aber der Standort dieser Artefakte ist von größter Bedeutung

> [22] - Webbrowser sind so konzipiert, dass sie eine Vielzahl von Informationen über die Aktivitäten ihrer Benutzer aufzeichnen und speichern können. Dazu gehören Caching-Dateien, besuchte URLs, Suchbegriffe, Cookies und andere. - Diese Dateien werden auf dem lokalen Computer gespeichert und können von jeder Person, die denselben Computer verwendet, leicht aufgerufen und abgerufen werden. Dies macht es auch für forensische Prüfer relativ einfach, die Internet-Aktivitäten eines Verdächtigen in Fällen zu untersuchen, in denen fragwürdige Websites besucht oder kriminelle Handlungen über das Internet durchgeführt wurden.

> [3] ■ Bestimmte Datentypen aus HTTP-Protokoll-Transaktionen oder skriptgesteuerten Aktionen in HTML-Seiten werden separat im Dateisystem gespeichert und führen zu unterschiedlichen Datenbankeinträgen: Cookies, Web Storage und Indexed Database Storage.

Private Browsing Artefakte: > [1] 1. Änderungen, die von einer Website ohne jegliche Benutzerinteraktion initiiert werden. Beispiele hierfür sind das Setzen eines Cookies, das Hinzufügen eines Eintrags zur Verlaufsdatei und das Hinzufügen von Daten zum Browser-Cache. 2. Änderungen, die von einer Website

initiiert werden, aber eine Benutzerinteraktion erfordern. Beispiele hierfür sind das Generieren eines Client-Zertifikats oder das Hinzufügen eines Passworts zur Passwortdatenbank. 3. Änderungen, die vom Benutzer initiiert werden. Zum Beispiel das Erstellen eines Bookmarks oder das Herunterladen einer Datei. 4. Nicht benutzerspezifische Zustandsänderungen, wie das Installieren eines Browser-Patches oder das Aktualisieren der Phishing-Blockierungsliste. ■ "geschützte Aktionen- Browser Artefakt, dass beim Verlassen des privaten Surfens gelöscht werden muss

Wie entstehen "Leckagen" von privaten Browsing Artefakten? [10] 1. Ein Fehler im Design und in der Entwicklung des Browsers 2. Das Betriebssystem übernimmt mehr Kontrolle über den Browser als es sollte, was dazu führt, dass Daten von außen abgegriffen werden

Common Locations: > Ort der Browserartefakte ("common locations") ausführlich beschrieben in: [4]

> [11] ■ Die Artefakte von Webbrowsern werden in bestimmten Ordnern im Betriebssystem gespeichert. ■ Die genaue Lage variiert je nach Browser, die Dateiformate bleiben jedoch gleich. ■ Es ist wichtig zu wissen, wo die Dateien gespeichert sind, um sie während des normalen und privaten Browsing-Modus untersuchen zu können. ■ Tabelle 6 zeigt die Standorte der Artefakte von Google Chrome wie Verlauf, Caches und Cookies. ■ Tabelle 7 stellt die häufigsten Standorte von Firefox-Artefakten wie Cookies, Cache, Verlauf und Lesezeichen vor. ■ Alle Änderungen in Firefox, wie Lesezeichen, installierte Erweiterungen und gespeicherte Passwörter, werden im Profildrucker gespeichert. ■ Wie in der Tabelle gezeigt, werden Cookies in cookies.sqlite gespeichert, während Cache-Dateien im cache2-Ordner zu finden sind. ■ Alle heruntergeladenen Lesezeichen, Dateien und der Verlauf werden in places.sqlite gespeichert. ■ Mögliche Informationen, die aus der Browser-Forensik extrahiert werden können, sind Browsing-Verlauf, Cache, Cookies, Lesezeichen und Download-Liste.

> [21] ■ Digitale Beweise in einem Webbrowser umfassen mindestens Caches, Verlauf, Cookies, Download-Dateilisten und Sitzungen. ■ Zumindest ein Minimum an digitalen Beweisen aus einem Webbrowser ist sehr wichtig und wird von Ermittlern genutzt, um einen Fall bei Internetnutzung zu analysieren.

Gründe für Browser-Artefakte bei Private Browsing: [10] > Fehler im Design und Entwicklung des Browsers -> führt dazu, dass Daten von innen nach außen durchsickern, d. h. Browser ist schuld > Betriebssystem übernimmt mehr Kontrolle über den Browser als es sollte, was dazu führt, dass Daten von außen abgegriffen werden, d. h. Betriebssystem ist schuld

Definition private Browsing Artefakt: =====  
Strings, die Aktionen des Browsing-Protokolls zugeordnet werden können: Keywords, URLs, HTML-Fragmente, E-Mail-Adressen, Betreffzeilen etc.

Warum Computer-Forensik: [12] ■ Die Untersuchung von digitalen Beweisen ist von großer Bedeutung, um Straftäter zu identifizieren und zur Rechenschaft zu ziehen.

Definition digitale Forensik [11] ■ Digitale Forensik konzentriert sich auf die Wiederherstellung von Speichermedien, um digitale Beweise für Cybercrime-Untersuchungen zu sammeln. ■ Die gewonnenen Beweise müssen jedoch in ihrem Originalzustand erhalten bleiben, um vor Gericht zulässig zu sein. ■ Der Prozess der Erwerbung, Untersuchung, Analyse und Berichterstattung von digitalen Beweisen muss forensisch einwandfrei durchgeführt werden. ■ Daher müssen Ermittlungsteams sich an die Phasen der digitalen Forensik halten, die auf weit verbreiteten Standards basieren. ■ Digitale Forensik-Investigatoren verlassen sich auf die Artefakte, die aus diesen Browser-Records auf dem Gerät zurückbleiben, und

verwenden forensische Techniken, um die Artefakte zu erfassen, um Beweismittel zu finden. ■ Die Artefakte werden im Computer-Speicher gespeichert, nachdem alle Browser-Verläufe, Caches und Cookies gelöscht wurden, was es für digitale forensische Gutachter einfach macht, die Daten zu extrahieren.

Definition Browser Forensics > [12] ■ Web-Browser-Forensik sammelt und identifiziert Beweise und Informationen im Zusammenhang mit einem Verbrechen aus wiederhergestellten Browser-Sitzungen - Forensische Analyse des Webbrowsers beinhaltet die Wiederherstellung von Browsing-Artefakten, die Informationen über die Online-Aktivitäten eines Verdächtigen offenbaren. - Browser-Forensik wird für Ermittler immer wichtiger, da Suchverlauf, Download-Aktivität und Seitenaufrufe das Verständnis für das kriminelle Motiv verbessern können.

Ziel digitale Forensik [11] ■ Digitale Forensik hat das Ziel, verwendbare Beweise für Computerkriminalität zu sammeln. ■ Cyberkriminalität wie Hacking, betrügerische Transaktionen und Diebstahl geistigen Eigentums erhöhen den Bedarf an digitaler Forensik, um auf Cyberkriminalität mit einem digitalen Gerät zu reagieren. (2022) A Comparative Analysis of Residual Data

Live-Forensik: unterschiedliche Definitionen in Literatur > Live-Forensik als "moderner Trend" der Computer-Forensik [6] Im Gegensatz zur traditionellen (toten) digitalen Forensik wird bei der Live-Forensik versucht, flüchtige Daten aufzubewahren und Gegenmaßnahmen für verschlüsselte Dateien auf einem Live-System zu ergreifen. > [8]: TODO! > [11] ■ „Live Forensics“ wird auch als „Live System Acquisition“ bezeichnet. ■ Diese Methode wird angewendet, wenn das System in Betrieb ist, um potenzielle Artefakte im flüchtigen Arbeitsspeicher (RAM) zu finden, die als Beweismittel genutzt werden können. ■ Viele Spuren von Computer-Sitzungen und Artefakte sind nur im flüchtigen Speicher zu finden und können nicht von externem Speicher aus ausgelesen werden. ■ Die Daten können jedoch nicht gesammelt werden, da sie verloren gehen, sobald das System gestoppt oder neu gestartet wird. ■ Die RAM-Daten müssen daher mit besonderen Verfahren behandelt werden, um ihre Integrität und Zuverlässigkeit während der Analyse zu gewährleisten. ■ „Live Forensics“ ist nützlich, um auch Ereignisse zu untersuchen, die nur während der Nutzung des Systems aufgetreten sind, und um Daten effizient im flüchtigen Arbeitsspeicher zu speichern. ■ Digitale Forensik kann dazu genutzt werden, die Gültigkeit von Beweismitteln bei Gerichtsverfahren zu untersuchen. ■ Nach der Identifikation und Sammlung von potenziellen Beweismitteln wird in den meisten Fällen eine exakte Kopie der Daten erstellt, um sie als Backup zu nutzen und Veränderungen zu vermeiden. ■ Es gibt zwei Arten von forensischen Techniken, um Speicherabbilder zu erstellen: „Dead Forensics“ und „Live Forensics“. ■ Bei „Live Forensics“ hingegen wird das System im laufenden Betrieb untersucht, was oft schwieriger ist, aber auch wertvolle Informationen liefern kann. > [21] ■ Forensische Untersuchung eines Systems, während es in Betrieb ist ■ Daten gehen verloren, wenn das System heruntergefahren oder neu gestartet wird ■ Verwendung bei flüchtigem Speicher wie RAM ■ RAM-Erfassung durch RAM-Forensik-Tool ■ Ziel ist es, den normalen Betrieb des Systems nicht zu beeinträchtigen ■ Live Forensics liefert wichtige Informationen für die Analyse ■ Analyse von digitalen Beweisen aus dem RAM mit Memory Analysis Tool. ■ Eine Lösung für dieses Problem ist die Live-Forensik, um Daten aus dem Arbeitsspeicher zu extrahieren, bevor sie gelöscht werden. ■ Diese Forschungsmethode wird verwendet, um Webbrowser im Allgemeinen und insbesondere tragbare Webbrowser zu analysieren.

Beispiele Live-Forensik > [24] ■ Volatiler Speicher (Memory Inspection) kann eine wichtige Informationsquelle für forensische Untersuchungen sein ■ DNS-Caching ist eine Bedrohung für private Browsing: Diese Schwachstelle entsteht, weil das Betriebssystem DNS-Anfragen des Browsers im Cache speichert, unabhängig davon, ob der Browser im privaten Modus ist oder nicht > [12] ■ Registry

Snapshots: Um Veränderungen im System-Registry aufgrund der Browserinstallation zu verfolgen, wurde Regshot verwendet, um vor der Installation einen Snapshot der Registry aufzunehmen. - Ein zweiter Snapshot wurde nach der Installation des Browsers aufgenommen und mit dem ersten verglichen. - Regshot generiert einen Bericht über die Ergebnisse, der die neuen Dateien und Ordner zeigt, die dem Registry-Schlüssel hinzugefügt wurden.

Vorteile Live-Forensik > In Literatur bekannt: Die meisten Informationen im RAM > [10] ■ Die meisten Daten können in den RAM-Speichergeräten des Betriebssystems gefunden werden. > [15] ■ Da es wahrscheinlich ist, dass RAM-Aufnahmen Inhalte der Browsing-Session (z.B. durch Caching) aufzeigen, wurde dies in das Projekt aufgenommen, insbesondere da Warren (2017) dies aufgrund von Zeitbeschränkungen nicht tun konnte. > [15] ■ Live Analyse während der Laufzeit einer Anwendung ist besonders vorteilhaft, um zu verstehen, wie das Betriebssystem und die Anwendung interagieren. ■ Live Analyse kann potenziell mehr Informationen zur Browsing-Session liefern, da die Designbemühungen des Tor-Projekts darauf abzielten, Schreibzugriffe auf die Festplatte zu vermeiden.

Herausforderungen von Live-Forensik = Kontaminieren von Beweismitteln [6] Die größten Herausforderungen während des Datenerfassungsprozesses sind: Datenveränderung und Abhängigkeit vom Betriebssystem des verdächtigen Systems; wenn der Erfassungsprozess die Daten verändert, werden die Gerichte die Daten als forensisch untauglich abweisen

Definition Dead Forensik: > [11] ■ Bei „Dead Forensics“ wird der Computer oder das Gerät, das untersucht werden soll, zuerst heruntergefahren, bevor das Speicherabbild erstellt wird. > [10] - Physische Speichererfassung ist nicht übliche Praxis und in den meisten Fällen nicht verfügbar > [8]: TODO! > [12] - Oft einzige Option: Analysen von Festplatten-Images von ausgeschalteten Geräten - Gründe für „einzige Option“: o Verzögerungen bei der Bearbeitung o Personalmangel bei den forensischen Untersuchern - also unrealistisch und unpraktisch, beschlagnahmte Geräte eingeschaltet zu lassen. - Ausschalten eines Geräts reduziert Risiko einer Datenänderung (versehentlich oder absichtlich) - isoliert es vom Netzwerk, um etwaige Versuche, es ferngesteuert zu löschen, zu verhindern, unter anderem. > [11] -> widersprüchlich? ■ System wird heruntergefahren, bevor das Speicherabbild erstellt wird. ■ Volatile Dateien gehen verloren: versteckte Dateien, ausgetauschte Dateien, Web-Aktivitäten, Artefakte und Log-Dateien ■ Das Ziel ist es, eine genaue Kopie des nichtflüchtigen Speichers zu erstellen, bevor das System heruntergefahren wird, um die Originalität der Beweismittel zu erhalten.

Beispiele Dead Forensik: > Stichwortsuche in Festplatten-Images nach herunterfahren [24] > Timestamps von Dateien [24] > SQLite Datenbanken [24] > Unallocated Space [24] > Registry-Hives auf Festplatte, z.B. NTUSER.DAT [24]

Probleme bei Dead Forensik > [6]: TODO!

Wann Live-, wann Dead Forensik? [11] ■ Die Wahl der Methode hängt von der Art der Untersuchung und der verfügbaren Zeit ab. ■ Das Ziel ist es, eine genaue Kopie des Speichers zu erstellen, um die Integrität der Beweise zu bewahren und das Risiko von Veränderungen zu minimieren.

Definition: Darknet Forensik: [20] ■ Motivation Darknet Forensik: o Terroristen, Kriminelle, extremistische Gruppen und Hassorganisationen nutzen das Darknet, um Cybercrime zu begehen. o Die Verwendung von TOR und Bitcoin auf dem Darknet erschwert die Verfolgung von Straftaten durch digitale Forensik-Experten. o Die vorgeschlagenen forensischen Techniken können digitale Forensik-Experten helfen, mit Cybercrime-Fällen im Zusammenhang mit dem Darknet umzugehen. ■

Darknet-Forensik sind in zwei Kategorien unterteilt: 1. TOR-Browser-Forensik: ■ vier Möglichkeiten zur Extraktion von TOR-Browser Artefakten: RAM-Forensik, Registry-Änderungen, Netzwerk-Forensik und Datenbank 2. Bitcoin-Transaktions-Forensik: Extrahieren von forensischen Artefakten aus Bitcoin-Wallet-Anwendung

### 3 Ziel der Arbeit

Wichtig: White-Box Ansatz gemäß local Attacker in [1] - Das Ziel des Angreifers besteht darin, für eine bestimmte Menge von HTTP-Anfragen, die er wählt, festzustellen, ob der Browser eine dieser Anfragen im privaten Browsing-Modus ausgeführt hat oder nicht. Wenn der lokale Angreifer dieses Ziel nicht erreichen kann, gilt die Implementierung des privaten Browsers als sicher. - Local Attacker weiß, wonach er sucht!

Forensiker müssen Funktionsweise von Private Browsing kennen [10] ■ Die Kenntnis der Erfolgsrate der PB-Technologie unterstützt die Strafverfolgungsbehörden bei digitalen Untersuchungen von Internetinhalten ■ Internetbeweise sind oft entscheidend für Untersuchungen ■ Bestimmung des Umfangs und des Erfolgs von PB-Technologie unterstützt die Strafverfolgungsbehörden bei digitalen Untersuchungen von Internetinhalten ■ Durch die Bestimmung des Umfangs und des Erfolgs von PB-Technologie können sie unnötige Datenverarbeitung und Zeitverschwendung vermeiden, die Untersuchungseffizienz verbessern und sicherstellen, dass keine wichtigen Inhalte übersehen werden. Daher können diese Punkte dazu beitragen, die Effektivität und Effizienz von Untersuchungen zu verbessern, insbesondere in Fällen, in denen Vor-Ort-Triage stattfindet oder in denen eine SHPO angeordnet wurde. Drei Punkte wichtig: ■ Wenn der Verdacht besteht, dass PB stattgefunden hat, hilft es zu wissen, wie erfolgreich die PB-Funktion eines bestimmten Browsers ist, um unnötige Datenverarbeitung (und Zeitverschwendung) zu vermeiden, wenn tatsächlich keine Browserdaten auf einem Gerät vorhanden sind. ■ Die Kenntnis darüber, wo PB möglicherweise Informationen zu Browsing-Sitzungen preisgibt, verbessert die Effizienz von Untersuchungen und verhindert, dass wichtige Inhalte übersehen werden. Dies ist besonders wichtig bei Vor-Ort-Triage, wie sie in einigen Fällen mit einer SHPO angeordnet wird.

Ziel der Arbeit: ===== - Welche Browsing Artefakte werden beim private Browsing auf einem Rechner hinterlassen, welche zeigen, dass eine Browsing Aktion vom Browser durchgeführt wurde? - Das heißt: o Es wird nach Browsing Artefakten gesucht, welche die Zuordnung „Durchgeführte Browsing Aktion“ <-> Browser ermöglichen o Vor, während und nach private Browsing Session nach Browsing Artefakten suchen, welche dem Browser zugeordnet werden können - Negativbeispiel: Suche in Hexdump nach im Browser gesuchtem String nicht als Beweis ausreichend, dass private Browsing Artefakte gefunden wurde. - Kategorisierung nach [18]: > Browsing History > Usernames/Email Accounts > Images

=> Thematisiert in [18]: o It appeared that the overall best way to recover residual data was to obtain the evidence from RAM or working memory, o Kritik: Oft nur String Match in RAM-Hex als Nachweis für PB genannt -> ausreichend? (Evtl. Gegenexperiment mit Editor)

Warum muss String-Artefakt Browser zugeordnet werden können? [11] ■ Die Artefakte, die von den Browsing-Aktivitäten eines Kriminellen zurückgelassen wurden, können mit forensischen Tools extrahiert werden, um die Untersuchung des Ermittlers zu unterstützen. ■ Die erlangten Beweise müssen vor Gericht zugelassen werden, insbesondere digitale Beweise, da sie ohne ordnungsgemäße Verfahren leicht manipuliert werden können. ■ Es gibt bestimmte Merkmale von digitalen Beweisen, die Gerichte

nach folgenden Kriterien akzeptieren: 1. Durchsuchungsbefehle - Beweise, die ohne Genehmigung erlangt wurden, können vor Gericht nicht anerkannt werden. 2. Berichte - Alle Prozesse, Werkzeuge, Methoden, Techniken, spezifischen Zeit- und Datumsangaben sowie die Beweiskette müssen formell dokumentiert werden, um die Authentizität der digitalen Beweise vor Gericht zu demonstrieren und zu unterstützen. 3. Beweisauthentifizierung - Der ursprünglich erhaltene Beweis sollte durch Vergleich der Hash-Werte mit dem Kopiebeweis übereinstimmen. Der erworbene Beweis muss unverändert bleiben, um die Gerichte mit genauen Informationen zu überzeugen. Gerichte akzeptieren Kopien von Beweisen, wenn der ursprüngliche Beweis verloren gegangen oder zerstört wurde, die Kopie jedoch noch intakt ist.

Ziele anderer Arbeiten: ===== > [11] - Die Art der extrahierbaren Daten zu untersuchen - den Unterschied zwischen privatem und normalem Surfen zu vergleichen - zu analysieren, welcher Browser die vollständigeren residualen Daten liefert. > [14] ■ ob bestimmte Arten von Browser-Daten gefunden werden konnten (Webseiten, Verlauf, Download-Verlauf, besuchte URLs und Suchbegriffe) > [21] ■ Das Ziel dieser Studie ist es, eine Rahmenbedingung für die Analysephasen des Webbrowsers im privaten Modus und Anti-Forensik vorzuschlagen, um eine effektive und effiziente forensische Untersuchung zu ermöglichen. ■ Die Studie nutzt Live-Forensik, um detailliertere Informationen über den Computer zu erhalten, während er noch in Betrieb ist, und eignet sich daher besser für die schnelle Datenerfassung in Echtzeit. > [24] ■ umfassende Analyse der privaten Browsing-Funktion in den vier beliebtesten Webbrowsers (IE, Firefox, Chrome und Safari) vorgestellt. > [11] - digitalen Forensikern helfen, Artefakte von Geräten zu verfolgen, die Live-Memory-Erfassung verwenden > [15] - Methodik entwerfen, um folgende Fragen zu beantworten: 1. Kann Tor den Benutzer schützen, indem es Beweise für dessen Nutzung aus dem RAM löscht, wenn die Browsing-Sitzung geschlossen wird? 2. Kann die Tor-Nutzung zu vier Schlüsselmomenten erkannt werden: während das Browser-Fenster geöffnet ist, nach Schließen des Browser-Fensters, nach dem Löschen des Installationsverzeichnis/zugehöriger Dateien und nach dem Ausloggen des Benutzers? 3. Können Dateien aus dem Browsing-Protokoll in der Live-Forensik mit Tor 7.5.2 wiederhergestellt werden, der zum Zeitpunkt der Schreibens aktuellsten Version? - Die Experimente wurden im mobilen Modus mit Tor wiederholt, d.h. von einem USB-Stick ausgeführt. (!!!) zu bestätigen, dass die Existenz und Nutzung des Tor-Browsers in Windows 10 nachweisbar ist. (!!!) nachweisen, dass Artefakte des Tor-Browsing-Protokolls auf dem Zielcomputer wiederhergestellt werden können. > [11] > In dieser Studie werden die residualen Daten zwischen Google Chrome und Mozilla Firefox Webbrowsers im normalen und privaten Browsermodus mithilfe eines forensischen Tools analysiert und verglichen. > [14] ■ In dem Projekt wurden die Effektivität der "privaten"Modus von vier weit verbreiteten Webbrowsers analysiert. > [24] ■ Ziel: Bewertung der Sicherheit des privaten Surfens in den Browsern Chrome, Safari, Firefox und IE ■ Die Autoren haben eine umfassende forensische Analyse durchgeführt, die sowohl Live-Memory-Analyse als auch Post-Mortem-Analyse umfasste. > [14] ■ Vier getestet: Firefox, IE, Safari und Chrome

Keine Ziele der Arbeit: ===== - zeitl. Kontext nicht wichtig ( Die zeitliche Reihenfolge innerhalb einer Logfile wird nicht berücksichtigt) - Private Browsing Indicators": Entering/Leaving Private Browsing [18] - Zeigen, dass ein Browser gestartet/geschlossen wurde - Zeigen, dass ein Browser im privaten Modus gestartet wurde - Zeigen, wann ein Browser gestartet/geschlossen wurde - Browser-Erweiterungen: [24] > Browser-Erweiterungen und ihre Auswirkungen auf das private Surfen wurden in einer Studie von Aggarwal et al. Im Jahr 2010 untersucht. -> Siehe Punkt „Add-Ons als Leck“ > Die Chrome-Erweiterung „Incognito Inspector“ kann im privaten Modus genutzt werden, um detaillierte Informationen über die Nutzeraktivitäten zu sammeln und in Echtzeit an einen Remote-Server zu senden. > Firefox-Erweiterungen sind standardmäßig im privaten Modus aktiviert und können



genutzt werden, um Nutzeraktivitäten aufzuzeichnen. > Internet Explorer-Erweiterungen sind in der Regel deaktiviert und erfordern die manuelle Aktivierung im privaten Modus. Die von den Autoren entwickelte Erweiterung funktionierte jedoch nicht, da sie aufgrund eingeschränkter Privilegien nicht auf die BHO-Klasse zugreifen konnte - [1] > Unterschiedliche Handhabung durch Browser: Gefährliche Leckage für private Browsing Artefakte > Entwickler von Add-Ons haben möglicherweise den privaten Browsing-Modus bei der Entwicklung ihrer Software nicht berücksichtigt, und ihr Quellcode wird nicht derselben rigorosen Überprüfung unterzogen wie die Browser selbst. > Gegenmaßnahme: [1] ■ Es wurde eine Firefox-Erweiterung namens ExtensionBlocker entwickelt, um unsichere Erweiterungen im privaten Modus zu blockieren

## 4 Methodik

Nachfolgend wird die vollständige Vorgehensweise der Versuchsdurchführung erklärt. Dazu ist es notwendig eine Methodik zu definieren.

In der Browserforensik ist eine definierte Methodik notwendig, um die Komplexität moderner Browser zu bewältigen. Sie bildet eine wissenschaftliche Basis für den durchgeführten Versuch sowie einen Leitfaden für Ermittler bei zukünftigen Untersuchungen. [1, 10, 11] Ein oft verwendetes Vorgehensmodell in der Computer Forensik ist das "Generic Model Computer Forensics Investigations", kurz GCFIM. [25] Ähnlich zum abschnittsbasierter Verlauf einer forensischen Untersuchung des Bundesamt für Sicherheit in der Informationstechnik ist es in Phasen mit definierten Abläufen gegliedert.

Izzati et al. haben diese Phasen auf Browserforensik abgebildet: [11]

- Vorbereitung: Versuchsplanung und Konfiguration der Versuchsumgebung.
- Datensammlung: Speicherabbilder identifizieren und während des Browsing Szenarios erstellen.
- Datenanalyse: Suche nach Browsing Artefakten in gesammelten Daten.
- Dokumentation: Vorgehensweise und gefundene Artefakte dokumentieren.

Die Dokumentationsphase entspricht in dieser Arbeit Kapitel "Vergleich der Browser"(TODO!). Die Methodik der anderen Phasen wird nachfolgend beschrieben.

### 4.1 Vorbereitung

In der Vorbereitungsphase wird der durchgeführte Versuch geplant sowie die Versuchsumgebung konfiguriert. [11] Versuchsplanung zählt die Auswahl von Browsern und Tools sowie die Definition eines durchzuführenden Browsing-Szenarios. Die Konfiguration der Versuchsumgebung umfasst die Installation und Konfiguration der notwendigen Software und Hardware.

#### 4.1.1 Browserauswahl

Für diese Arbeit dazu entschieden: zwei weit verbreitete Brower verwenden + zwei Browser mit verstärktem Schutz der Privatsphäre.

Laut Statistik "Global market share held by leading internet browsers from January 2012 to May 2023"(Stand 23. Mai 2023) von Statista ist Chrome mit 62,82% der meistgenutzte Browser weltweit. Danach folgen Safari (20,86%), Edge (5,28%) und Firefox (2,77%).

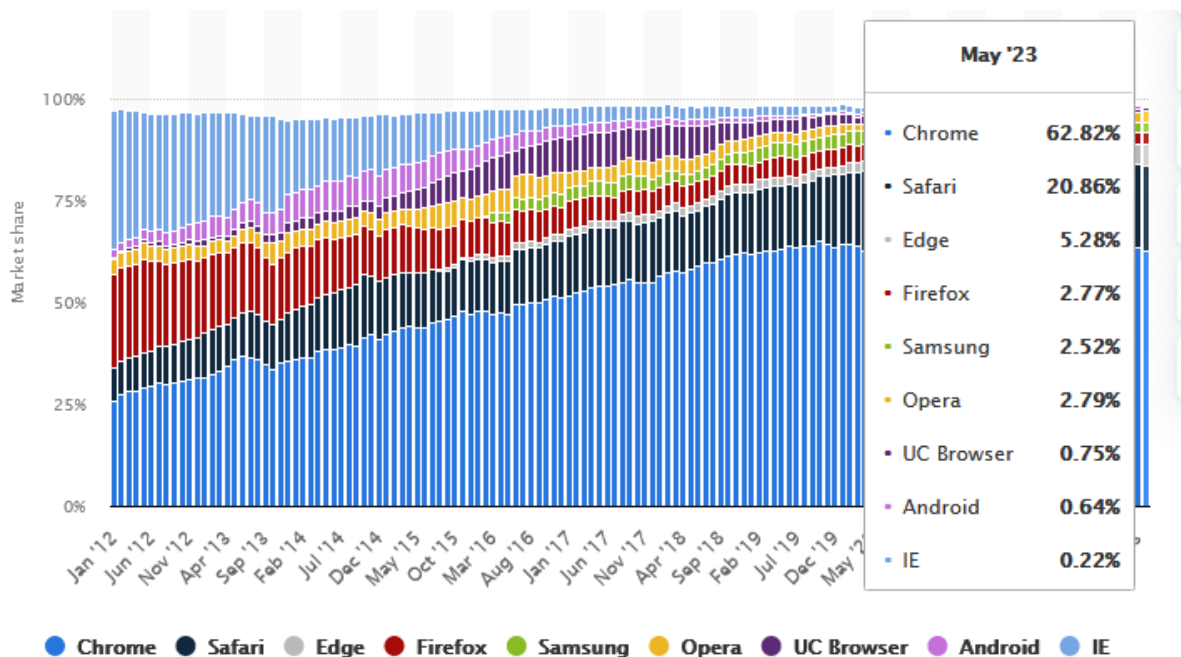


Abbildung 4.1: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

Problem: Safari hauptsächlich auf Mac OS genutzt, für Windows nur Safari 5.1.7 oder ältere Versionen. Weiteres Problem: In Literatur zwar oft Internet Explorer (TODO: Quellen) untersucht, den Nachfolger Edge wurde bisher jedoch kaum in Literatur betrachtet. -> Nur 3 von 23 untersuchten Papern nahmen Edge mit in die Lister der analysierten Browser auf: [4, 5, 10] -> Ziel der Arbeit: keine neuen wissenschaftlichen Erkenntnisse.

Deshalb: wird neben Chrome noch Firefox als zweiter "regulärer" Browser aufgenommen: Sowohl Firefox als auch Chrome werden in 15 von 23 untersuchten Papern analysiert: [1, 4, 5, 10–18, 21, 22, 24]

Weiterhin werden zwei Browser mit verstärktem Schutz der Privatsphäre ausgewählt. Um die Browser mit den regulären Browsern vergleichen zu können, werden Browser gewählt, die auf den regulären Browsern basieren. Basierend auf Chromium wird der Browser "Brave" gewählt. Für Firefox wird der Tor-Browser gewählt, eine modifizierte Version von Firefox.

## Firefox

Der Browser Mozilla Firefox, kurz Firefox, ist ein open-source Webbrowser der gemeinnützigen Organisation Mozilla. Seit seiner Einführung im Jahr 2004 hat sich Firefox als beliebter Webbrowser etabliert.

Mozilla bewirbt den Firefox Browser mit seinem Fokus Privatsphäre und Sicherheit. Im Jahr 2009 führte Firefox den "privaten Modus" ein, der es Benutzern ermöglichte, das Surfen im Internet ohne Speicherung von Verlaufsdaten und Cookies zu genießen. Abbildung X und Y (TODO!) zeigen, die Aktivierung des privaten Modus über das Menü in der rechten oberen Fensterecke. Der private Modus öffnet sich anschließend in einem neuen Firefox-Fenster

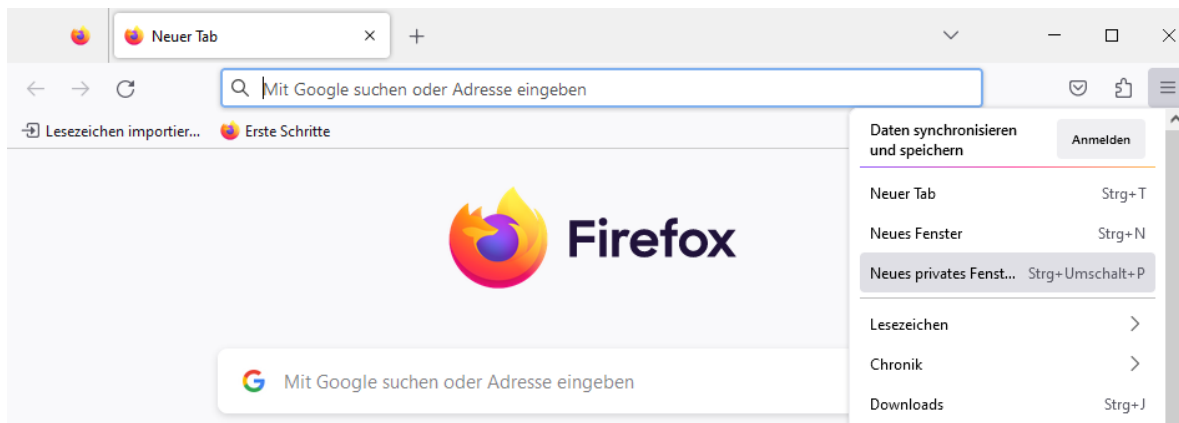


Abbildung 4.2: Private Mode Firefox 1

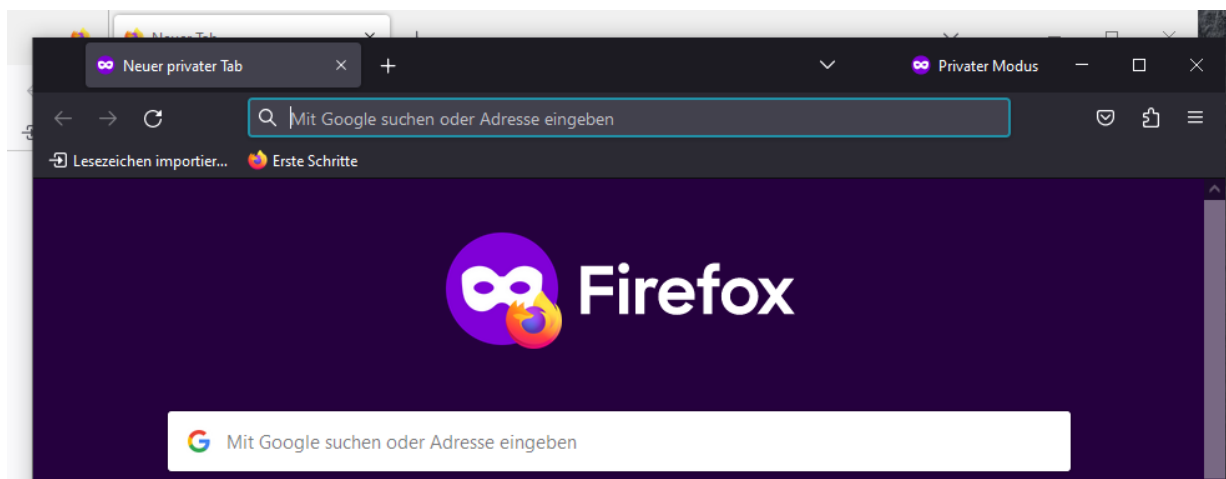


Abbildung 4.3: Private Mode Firefox 2

- Mozilla garantiert über den privaten Modus, dass Surfaktivitäten vor anderen Personen verborgen werden, die Firefox am selben Computer wie Sie verwenden. - Dazu zählt laut Mozilla: Passwörter, Chronik und Cookies gespeichert werden. Dies entspricht dem Schutz gegenüber des "Lokalen Angreifers", wie er in Kapitel X (TODO!) definiert ist. - Es wird ausdrücklich darauf hingewiesen, dass die besuchten Webseiten und Ihr Internetanbieter (ISP) weiterhin anhand Ihrer IP-Adresse Informationen über die von Ihnen besuchten Seiten sammeln, selbst wenn Sie nicht angemeldet sind.

Somit ist der private Modus von Firefox laut Mozilla vor dem lokalen Angreifer geschützt, jedoch nicht vor dem Webangreifer.

Für diesen Versuch: Firefox Version 112.0.2 (64 Bit)

## Tor

Der Tor Browser, früher Tor Browser Bundle, ist ein auf Firefox basierender Webbrowser, der das Tor-Netzwerk nutzt. - wird von der gemeinnützigen Organisation "The Tor Project" entwickelt - Das Tor-Netzwerk ist ein Netzwerk virtueller Tunnel, der den Datenverkehr über drei zufällige Server ("Relays") im Tor-Netzwerk leitet, bevor er über den letzten Server (Exit-Relay) ins öffentliche Internet gelangt. - Jedes Relay kennt nur den vorherigen und den nächsten Schritt des Datenverkehrs, aber nicht den gesamten Pfad oder die Quelle der Verbindung. - Dadurch soll Privatsphäre und Sicherheit im Internet geschützt und verbessert werden. - Das Tor-Netzwerk wird von einer dezentralen Community von Freiwilligen betrieben und verwaltet - Der Tor Browser ermöglicht es Benutzern, Domains mit .onion als Top-Level-Domain zu besuchen. Die Domainnamen werden kryptografisch generiert und lassen nicht auf den Webseitenamen schließen.

Über Tor-Browser mit Tor-Netzwerk verbinden:

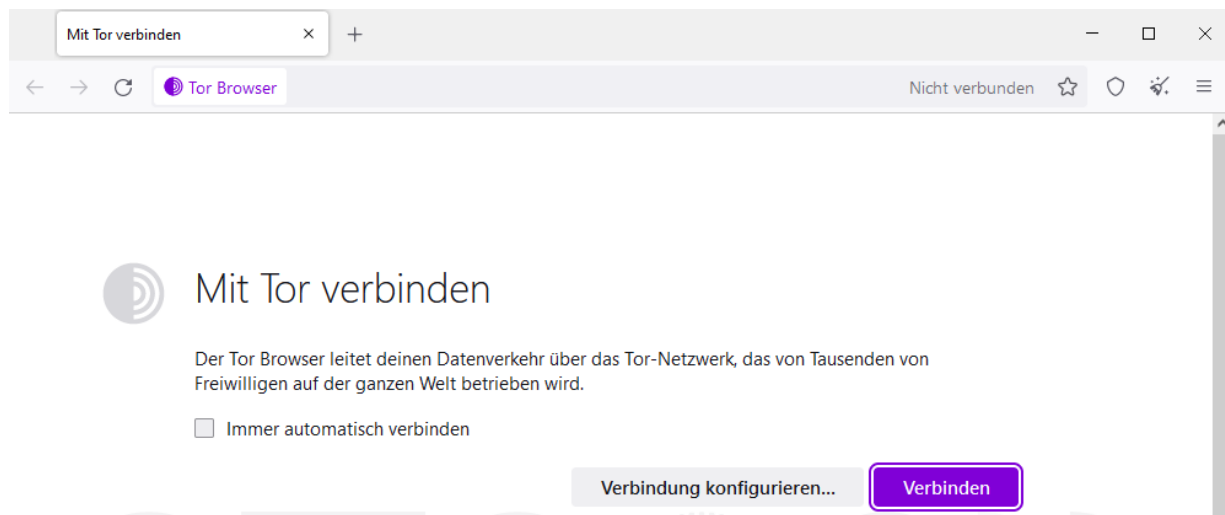


Abbildung 4.4: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

Der Tor Browser wirbt im Gegensatz zu Firefox mit folgendem Schutzmaßnahmen gegen Webangreifer:

- Internetdiensteanbieter können Internetaktivitäten aufgrund Natur des Tor-Netzwerks nicht verfolgen -
- Die Betreiber der besuchten Websites sehen eine Verbindung vom Tor-Netzwerk anstelle der echten IP-Adresse des Rechners - Kein "fingerprinting" = Nutzer anhand Browserkonfiguration identifizieren.

Der Tor Browser wirbt mit folgendem Schutzmaßnahmen gegen lokalen Angreifer:

- Schutz vor lokalem Angreifer durch Modifikation von Firefox: Tor Browser basiert auf dem Extended Support Release von Firefox. - Lokaler Angreifer: Tor Browser does not keep any browsing history. Cookies are only valid for a single session (until Tor Browser is exited or a New Identity is requested).
- Technische Umsetzung 1. Standardeinstellungen geändert 2. Zusätzliche Erweiterungen installiert: - Torbutton": Mit Tor verbinden > Screenshot - "NoScript": JavaScript und andere potenziell schädliche Inhalte nur von vertrauenswürdigen Websites Ihrer Wahl ausgeführt

Zusätzliche Funktion: "Neue Identität" Die Funktion "Neue Identität" im Tor Browser ermöglicht es,

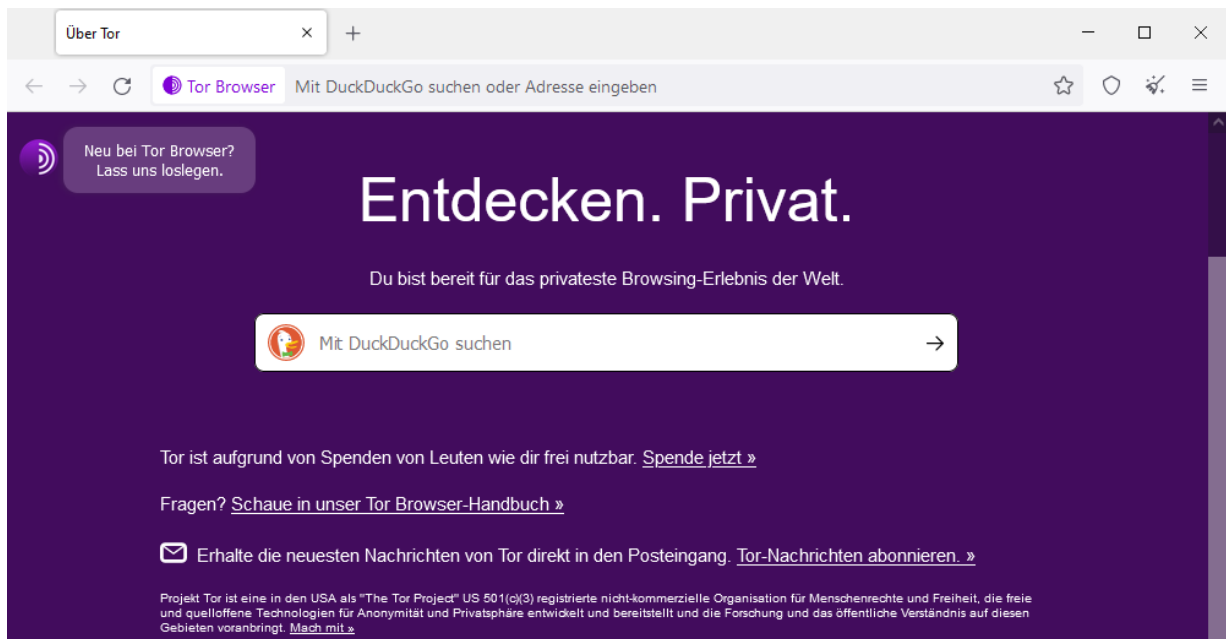


Abbildung 4.5: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

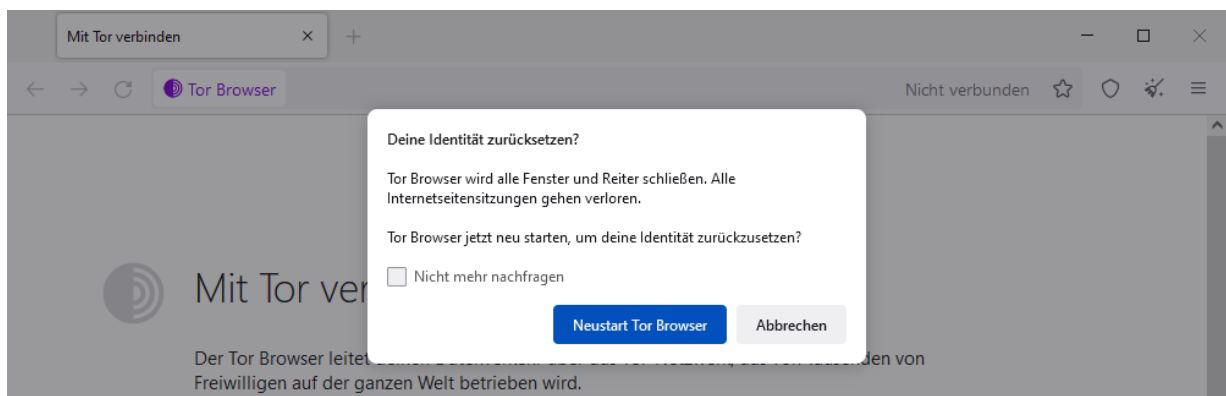


Abbildung 4.6: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

alle aktuellen Tabs und Fenster zu schließen, sämtliche private Informationen wie Cookies und Verlauf zu löschen und für alle Verbindungen neue Relays zu verwenden.

Für diesen Versuch: Tor Version 12.0.4 (64 Bit)

## Chrome

## Brave

### 4.1.2 Browsing Szenario

Siehe Ziel der Arbeit: Versuch soll nicht Situation eines Ermittlers simulieren, der eine unbekannte Datenlage hat. Stattdessen: Bereits vor Analyse bekannt, nach welchen Daten gesucht werden muss.

Deshalb wird für Versuch definiert, mit welchen Daten der Rechner kontaminiert werden soll.

Im Falle der Browser-Forensik wird dazu ein sog. "Browsing Szenario" festgelegt. (TODO: Quelle, alternative Namen).

Dabei handelt es sich um Reihe von fest definierten Aktivitäten, die für jeden zu untersuchenden Browser durchgeführt werden.

Anforderungen an Browsing Szenario - Ziel: PB Artefakte, die ausschließlich in Browsing-Szenario vorkommen, bspw. "twitteröder "facebook" bereits in vielen Windows-Standardanwendungen enthalten.  
- (TODO: Literatur)

Folgende Schritte werden in jedem Browser durchgeführt:

1. [www.google.com](http://www.google.com) aufrufen
  - 1.1. Alle Cookies akzeptieren
  - 1.2. Google-Suche nach "pfaffenhofen"
2. [www.google.com](http://www.google.com) aufrufen
  - 2.1. Cookies alle akzeptieren
  - 2.2. Google-Suche nach "nanoradar"
3. [www.google.com](http://www.google.com) aufrufen
  - 3.1. Cookies alle akzeptieren
  - 3.2. Google-Suche nach "mallofamerica"
  - 3.3. Auf Suchergebnis "mallofamerica.com" klicken
4. [www.google.com](http://www.google.com) aufrufen
  - 4.1. Cookies alle akzeptieren
  - 4.2. Google-Suche nach "mooserliesl"
  - 4.3. Auf Suchergebnis "mooserliesl.de" klicken
5. "[www.unitree.com](http://www.unitree.com)" über URL-Leiste öffnen
6. "[www.donaukurier.de](http://www.donaukurier.de)" über URL-Leiste öffnen
  - 6.1. Donaukurier Logo in neuem Tab öffnen

Kategorie	Private Browsing Artefakt	Schritt im Browsing Szenario
Suchbegriff	"pfaffenhofen"	1.2
	"nanoradar"	2.2
	"mallofamerica"	3.2
	"mooserliesl"	4.2
URL	"mooserliesl.de"	3.3
	"mallofamerica.com"	4.3
	"unitree.com"	5.
	"donaukurier.de"	6.
Bild	0x89 0x50 0x4E 0x47 ... (PNG als Hexadezimalwerte)	6.1
E-Mail	"computerforensikvl@gmail.com"	7.1.1
	"Vorlesung23!"	7.1.2
	"cas0597@thi.de"	7.2.1
	"chs3702@thi.de"	7.2.1

#### 7. "mail.google.com" über URL-Leiste öffnen

##### 7.1. Mit google Account anmelden:

7.1.1. E-Mail = "computerforensikvl@gmail.com"

7.1.2. Passwort = "Vorlesung23!"

##### 7.2. Neue E-Mail schreiben:

7.2.1. Empfänger: "cas0597@thi.de" und "chs3702@thi.de"

7.2.2. Betreff: "Betrefftext"

7.2.3. Mailinhalt: "Mailinhalt"

Aus diesem Browsing-Szenario lassen sich sogenannte "private Browsing Artefakte", kurz PB Artefakte ableiten. Wie in Kapitel X (TODO!) beschrieben, handelt es sich dabei um Strings oder reguläre Ausdrücke, die eindeutig einem Schritt im Browsing-Szenario zugeordnet werden können. PB Artefakte in Tabell X (TODO!) beschrieben.

Bemerkung: "mail.google.com" nicht aufgeführt, da festgestellt, dass URL bereits in vielen Windows Standardanwendungen enthalten ist.

## VM Konfiguration

Best Practice in Browser Forensik: Versuche sowie Analysen in virtualisierter Umgebung durchführen. (TODO: Quellen) - Dadurch Reproduzierbarkeit der Ergebnisse sichergestellt - Keine Vermischung der PB Artefakte, wenn gleicher Rechner verwendet - Keine Vermischung der Versuchsumgebung mit der Analyseumgebung - Ergebnisse sind transportabel -> Zustände von Virtuellen Maschinen exportierbar - Deshalb oft in Literatur empfohlen



Als Virtualisierungssoftware für Versuch verwendet: Kostenlose Oracle VirtualBox VM, Version 7.0.8 r156879 (Qt5.15.2)

In Literatur zu Browserforensik empfohlen: pro Browser eine VM erstellen, auf der Browsing Szenario durchgeführt wird.

Dabei zunächst eine Benchmark-VM erstellt. Wird nach Basiskonfiguration als OVA exportiert und für jeden Browser dupliziert. Anschließend für jede VM entsprechenden Browser installiert.

**Betriebssystem** VM Betriebssystem: In den 23 untersuchten wissenschaftlichen Veröffentlichungen wurden die privaten Browsermodi ausschließlich unter Windows untersucht. Da Ziel dieser Arbeit ist keine neuen wissenschaftlichen Ergebnisse: Für diesen Versuch Windows 10 verwendet. Deshalb auf VM installiert: Windows 10 Pro, Build: 19045.2006, nicht aktiviert

**Speicher** In Literatur keine Angaben über empfohlene Festplattengrößen. Deshalb an Microsoft Empfehlungen orientiert: VM erhält 30 GB (VDI-Format) Festplatte, kein SSD-Laufwerk In Literatur ebenfalls kaum Angaben über empfohlene RAM Größe gemacht: > Wichtige Entscheidung, da wie später in Kapitel X beschrieben, RAM-Größe Auswirkungen auf Ergebnisse hat > Rochmadi et al.: 2 GB [21] > Ohana et al.: 4 GB [18]: > Hier: mit 6 GB die maximal mit verfügbaren Speicherressourcen auf Analyse-Rechner mögliche Größe gewählt, um später Speicherabbilder des Arbeitsspeichers zu sichern.

**Netzwerkconfiguration** VM wurde mittels Netzwerkbrücke direkt mit dem physischen Netzwerk des Hostsystems verbunden. Somit erhält jede virtuelle Maschine eine eigene IP-Adresse im Netzwerk des Rechners, auf dem die VM läuft. Der Netzwerkadapter der VM wurde erst nach Browserinstallation aktiviert, um eine versehentliche Kontaminierung der VM zu vermeiden.

### Installierte Programme auf VM

Um Programme auf VM zu installieren: Gemeinsamer Ordner zwischen VM und Rechner eingerichtet, auf dem VM läuft. Ordner wird in VM als Netzwerklaufwerk angezeigt Grund: VM erst mit Beginn von Browsing Szenario mit Netzwerk und Internet verbinden. Deshalb: Programme mussten auf Rechner auf dem VM läuft heruntergeladen werden und über gemeinsamen Ordner auf VM transportiert werden.

**Browserinstallation** Zunächst Browser installiert. Folgende Installationsverzeichnisse verwendet:

#### Firefox C:

- Program Files
- Mozilla Firefox
- firefox.exe

#### Tor C:

- Program Files
- Tor Browser

Browser  
firefox.exe

**Chrome** \*\*\*TODO!\*\*\*

**Brave** \*\*\*TODO!\*\*\*

Weiterhin wurden zwei Werkzeuge der Sysinternal-Abteilung von Microsoft installiert: "Process Monitor" und "Process Explorer". Hintergrund: siehe Ziel der Arbeit: ersuch soll nicht Situation eines Ermittlers simulieren, der eine unbekannte Datenlage hat. Mithilfe der Tools soll Browserverhalten während des Szenarios aufgezeichnet und untersucht werden können.

**Process Monitor** Process Monitor ermöglicht die Aufzeichnung aller Aktivitäten und Ereignisse, die auf einem Windows-System im Zusammenhang mit Prozessen, Dateisystemen, Registrierungseinträgen und Netzwerkverbindungen stattfinden. Die Aufzeichnung kann als "Process Monitor Logfile" (PML) exportiert werden. Die aufgezeichneten Aktivitäten können somit beliebig gefiltert werden, beispielsweise nach Prozessnamen oder Operation, die der Prozess durchgeführt hat. Die Möglichkeit zum Export als CSV-Datei ermöglicht eine umfassende Untersuchung mit weiteren Programmen, wie Microsoft Excel. Für diesen Versuch verwendet: Version 3.93

**Process Explorer** Weiterhin wurde "Process Explorer" installiert. Das Tool erweitert die Funktionen des Windows Task Managers und ermöglicht es, einen umfassenden Überblick über alle aktiven Prozesse und deren Eigenschaften zu erhalten. Beispielsweise können alle ausgeführten Windows Dienste mit ihren PIDs angezeigt werden. Für diesen Versuch verwendet: Version 17.04

### 4.1.3 Analysewerkzeuge

Neben Konfiguration der VM muss Analyseumgebung vorbereitet werden. Als Analyseumgebung dient der Rechner, auf dem die VM läuft. Spezifikationen: \*\*\* TODO \*\*\* Dazu: Diverse Tools zur Analyse installieren

#### Autopsy

Wie im nächsten Kapitel X (TODO!) beschrieben, müssen Festplattenabbilder untersucht werden. Dazu wird in der Literatur das Tool Autopsy empfohlen.

Autopsy ist ein Open-Source-Digital-Forensik-Tool, das auf der Sleuthkit-Bibliothek basiert. Sleuthkit ist eine Sammlung von Open-Source-Tools für die forensische Analyse von Dateisystemen. Es bietet Funktionen zum Lesen, Analysieren und Wiederherstellen von Daten aus verschiedenen Dateisystemen. Autopsy baut auf der Funktionalität von Sleuthkit auf und bietet eine grafische Benutzeroberfläche für die forensische Analyse. Wurde hauptsächlich in Java geschrieben. Es erweitert die Funktionalität von Sleuthkit, indem es zusätzliche Tools, Plug-Ins und Automatisierungsfunktionen bereitstellt, um den forensischen Untersuchungsprozess zu unterstützen.

Für diesen Versuch verwendet: Version 4.20.0

## Volatility

Neben der Analyse von Festplattenabbildern müssen gemäß Kapitel X Abbilder des Arbeitsspeichers untersucht werden. Dazu wird das in der Literatur empfohlene Tool "Volatility" verwendet.

Das Volatility Memory Forensics Framework ist ein Open-Source-Tool, das für die forensische Analyse von Arbeitsspeicherabbildern verwendet wird. Es ist speziell darauf ausgerichtet, Informationen und Artefakte aus dem physischen oder virtuellen Arbeitsspeicher eines Computers zu extrahieren.

Geschrieben in Python, frei auf GitHub verfügbar: Version: Volatility3, Version 2.4.1 (aktuellster Release)

Für diesen Versuch verwendet: "Volatility3- vollständige Neuschreibung des Volatility Memory Forensics Frameworks, die im Jahr 2020 veröffentlicht wurde. Behebt technische und Performanceprobleme der vorherigen Version.

Oft beworbener Vorteil von Volatility3: kein "Profile" mehr notwendig. Volatility3 erstellt Symboltabellen für Windows-Speicherabbilder basierend auf dem Speicherabbild selbst. Dabei handelt es sich um eine Konfigurationseinstellung, welche die Speicherstruktur und die Verhaltensweisen des Betriebssystems definiert.

Volatility basiert auf Plugins, welche spezifische Funktionen und Analysen für verschiedene Aspekte des Systems bereitstellen. Für diesen Versuch werden folgende Plugins verwendet.

- pslist
- yarascan
- memmap
- filesca
- svcscan

Genaue Beschreibung der Plugins und deren Zusammenhang: Siehe Analysephase in Kapitel X (TODO!).

## Übersicht verwendete Software

Tabelle X listet zusammenfassend alle in diesem Versuch verwendeten Software-Programme, deren Zweck sowie Version auf.

Neben den Programmen zur Analyse der Speicherabbilder: diverse zusätzliche unterstützende Tools zur vollständigen Analyse benötigt, diese werden nicht genauer beschrieben. - Nur in Tabelle aufgenommen

Software	Zweck	Version
Oracle VirtualBox	Virtualisierung	7.0.8 r156879
Windows 10 Pro	VM Betriebssystem	Build: 19045.2006
Process Monitor	Aufzeichnung Prozessaktivitäten	3.93
Process Explorer	Darstellung der Eigenschaften aktueller Prozesse	17.04
Autopsy	Analyse Festplattenabbilder	4.20.0
Volatility	Analyse RAM-Abbilder	Volatility3 Version 2.4.1
HxD	Analyse Binärdateien in hexadezimaler und ASCII-Darstellung	2.5.0.0
Notepad++	Analyse strukturierter Dateiformate, z.B. JSON, XML	8.4.5
Registry Explorer	Grafische Oberfläche zur Untersuchung von Windows-Registry Hives	2.0.0.0
DB Browser for SQLite	Grafische Oberfläche zur Verwaltung und Untersuchung von SQLite-Datenbanken	3.12.2
sqldiff.exe	Befehlszeilen-Programm zur Anzeige von Unterschieden zwischen SQLite-Datenbanken	3.42.0
ChromeCacheView	Einlesen von Chrome Cache-Dateien und visuelle Aufbereitung des Inhalts	2.46
MZCacheView	Einlesen von Firefox Cache-Dateien und visuelle Aufbereitung des Inhalts	2.21
FirefoxCache2	Erweitert MZCacheView, um Firefox indexCachedatei zu analysieren	Commit b50ab4f
dejsonlz4	Dekomprimierung von .jsonlz4-Dateien	Commit c4305b8

## 4.2 Datensammlung

\*\*\* TODO: Schreiben, dass für diesen Versuch auch VM-Snapshots zu bestimmten Zeitpunkten aufgetaut werden können \*\*\*

In der Phase der Datensammlung werden alle potenziellen Beweismittel identifiziert, gesammelt. Ziel ist es, die Daten in einem Format zu sammeln, in dem sie in der nächsten Phase analysiert werden können. [11]

Für diesen Versuch umfasst dies die Durchführung des Browsing Szenarios sowie die Sammlung von Ressourcen, die potentielle private Browsing Artefakte enthalten.

### Process Monitor Logfiles

Gemäß dem in Kapitel X definierten Ziel der Arbeit wird der Versuch nicht aus Sicht eines Forensikers durchgeführt, der nur begrenzten Zugriff auf Beweismittel hat. Stattdessen soll mit diesem Versuch das Verhalten von privaten Browsingmodi möglichst vollständig untersucht werden, mit dem Ziel alle auf der VM hinterlassenen privaten Browsing Artefakte zu identifizieren. Den gleichen Ansatz verfolgten Fayyad-Kazan et al. [4] Sie schlagen deshalb vor, alle Aktivitäten des Browsers während Browsing-Szenarios aufzeichnen. Sie empfehlen dazu das oben in Kapitel X beschriebene Tool "Process Monitor". Mithilfe einer grafischen Oberfläche können die aufgezeichneten Aktivitäten als Process Monitor Logfile sowie CSV Datei gespeichert werden. Nach Empfehlung der Autoren liegt bei den Browseraktivitäten der Fokus auf Schreibaktivitäten im Dateisystem sowie geänderte Werte in der Registry. [4, 21] Um die aufgezeichneten PML-Dateien auf den Analyserechner zu transportieren, wird der bereits in Kapitel X zur Installation von Programmen verwendete gemeinsame Ordner verwendet.

### Speicherabbilder

Eine der Hauptaufgaben eines Computer-Forensischen-Ermittlers ist die Erstellung und Analyse von Speicherabbildern. [8] Dabei handelt es sich um eine direkte Kopie der Daten auf den Speichermedien des Rechners. [8] Dazu zählen beispielsweise Festplatten, der Arbeitsspeicher sowie angeschlossene Speichermedien wie USB-Sticks. Zur Identifikation von Artefakten auf dem untersuchten Rechner

muss vermieden werden, auf den originalen Speichermedien zu arbeiten. Dadurch können Beweise verfälscht werden und nicht mehr vor Gericht verwendet werden. [8] Aus diesem Grund muss es sich bei den analysierten Speichermedien stets um Abbilder (Images), also Kopien des originalen Mediums handeln.

Im Falle der Browser Forensik werden in der Literatur zwei Arten von Speichermedien untersucht: Festplatten (TODO: Alle Quellen) und der Arbeitsspeicher (TODO: Alle Quellen)

**Festplatten-Image** Um ein Abbild einer physikalische Festplatten zu erstellen gibt es diverse Software- und Hardware Lösungen. Da in diesem Versuch die Festplatten virtualisiert wurden, entspricht ein Festplatten-Abbild einem sogenannten "VM-Snapshot". Dabei handelt es sich um eine Momentaufnahme von einer virtuellen Maschine. Ein Snapshot erfasst den Arbeitsspeicher, den Prozessorzustand sowie den Festplatteninhalt zu einem bestimmten Zeitpunkt. Bei Oracle VirtualBox kann eine VM Snapshot über die grafische Oberfläche durchgeführt werden. Dazu wird ausgehend vom letzten beziehungsweise aktuellen Zustand der VM im Menü "Sicherungspunkte" die Option "Erzeugen" ausgewählt. Dies kann sowohl im an- und ausgeschaltetem Zustand der virtuellen Maschine durchgeführt

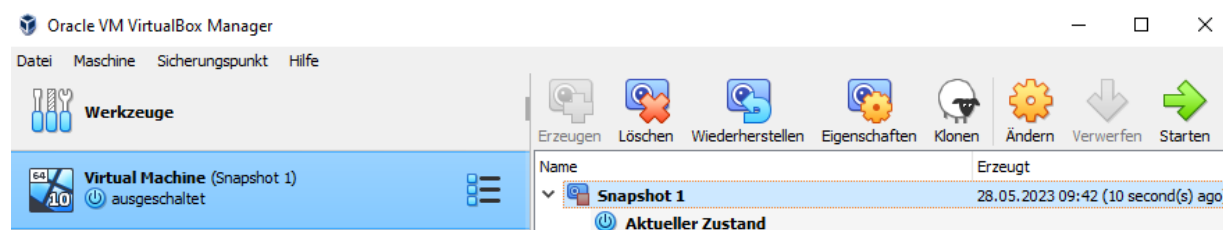


Abbildung 4.7: TODO

werden. Somit kann ein VM-Snapshot sowohl zur Live- als auch Post-Mortem-Forensik verwendet werden. Durch den Snapshot wird ein "Virtual Disk Image", eine VDI-Datei, im Snapshot-Ordner der VM erzeugt. Diese Laufwerksdatei enthält nur differentielle Daten zum vorherigen Snapshot bzw. zum aktuellen Zustand, wenn es sich um den ersten Snapshot handelt. Die Datei ist meist wenige KiloByte groß. Um aus den differentiellen Daten ein vollständiges Festplatten-Image zu erzeugen muss aus dem Snapshot über die Option "Klonen" eine eigenständige VM erstellt werden. Dabei muss die Option "vollständiger Klon" ausgewählt werden. Die VDI-Datei der geklonten VM enthält alle Festplatten-Daten zum Zeitpunkt des durchgeführten Snapshots.

Wie einleitend beschrieben, müssen die in dieser Phase gesammelten Daten in einem Format vorliegen, das in der Analysephase verwendet werden kann. Wie im nächsten Kapitel beschrieben, werden die Festplatten-Images mit dem Tool Autopsy analysiert. Da Autopsy kein VDI-Format unterstützt, müssen die Laufwerksdateien der geklonten Snapshots in ein das Image-Format (.img) umgewandelt werden. Dabei handelt es sich um ein generisches Dateiformat, welches ein binäres Abbild des Speichermediums speichert. Oracle VirtualBox bietet das in der Installation enthaltene Befehlszeilen-Tool "vboxmanage" für diese Dateiumwandlung an. Mit dem Befehl `vboxmanage clonehd <VDI_File>.vdi <IMG_File>.img -format raw` wird das VirtualBox Disk Image "VDI\_File" in die Image-Datei "IMG\_File" umgewandelt.

\*\*\* TODO: Hier Einlesen der Festplatten-Images => Zeitaufwändig! \*\*\*

**RAM-Dump** Wie oben beschrieben, enthält ein VM-Snapshot neben dem Festplatteninhalt den Inhalt des Arbeitsspeichers zum Zeitpunkt der Momentaufnahme. Ein sogenannter "RAM-Dump" fasst den genauen Zustand des Arbeitsspeichers, einschließlich der im Speicher befindlichen Daten, Programme und Prozesse. Der RAM-Dump eines Snapshots wird neben weiteren Daten in .sav Dateien gespeichert. VirtualBox bietet aktuell keine Möglichkeit aus diesen Dateien den RAM in einem analysierbaren Format zu extrahieren. Stattdessen wird von VirtualBox empfohlen, Abbilder des Arbeitsspeichers ebenfalls über das "vboxmanage"-Befehlszeilen-Tool durchzuführen. Im Unterschied zu Festplatten-Images, können RAM-Dumps ohne zusätzliche Hardware damit nur in angeschaltetem Zustand der virtuellen Maschine durchgeführt werden. Um ein Abbild des Arbeitsspeichers einer laufenden virtuellen Maschine zu erstellen wird folgender Befehl ausgeführt: `vboxmanage debugvm <VM Name> dumpvmmcore -filename <RAM Dump Dateiname>.elf`. Zur Analyse des RAM-Dumps ist keine weitere Verarbeitung notwendig. RAM-Dumps im .elf Format können direkt vom Analysetool Volatility eingelesen werden.

**Zeitpunkte zur Datensammlung** Wichtig für die Qualität der Versuchsergebnisse ist das Festlegen der Zeitpunkte im Browsing Szenario zum Sammeln der Daten. Dieses Thema wird in der Literatur kaum thematisiert. Die Autoren wählen die Zeitpunkte meist ohne Begründung. Ausschließlich Fayyad et al. geben an, dass sie die Process Monitor Logfiles während des Browsing Szenarios erstellen. Einen präzisen Start- und Endzeitpunkt der Aufnahme nennen sie nicht. Abbilder des Arbeitsspeichers werden bei 23 untersuchten Papern am häufigsten nach Durchführung des Browsing Szenarios und Schließen des Browsers durchgeführt [7, 11, 13, 18]. Mahlous et al. erstellen einen RAM-Dump mit geöffnetem Browser, jedoch keinen nach Schließen des Browsers [12]. Rochmadi et al. erstellen mehrere RAM-Dumps zu unterschiedlichen Zeitpunkten: Vor dem Schließen des Browser, nach dem Schließen des Browsers sowie nach dem Löschen der Registry [21]. Festplattenabbilder werden am häufigsten im Zuge der Post-Mortem-Forensik analysiert und nach Herunterfahren der VM gesichert [3–5, 10, 12–14, 18]. In einigen Fällen wurde gar nicht erwähnt, zu welchen Zeitpunkten im Browsing Szenario die Daten gesammelt wurden [1, 14, 16, 22–24].

Dieses Problem haben Muir, Leimich und Buchanan erkannt und Zeitpunkte zur Datensammlung vorgeschlagen, um das Browserverhalten während des Browsing-Szenarios vollständig zu überwachen und zu analysieren. Wie in Abbildung X (TODO!) dargestellt, wurde sich an diesen Zeitpunkten für diesen Versuch orientiert. Nach der Browser-Installation, vor Beginn des Browsing-Szenarios wird der

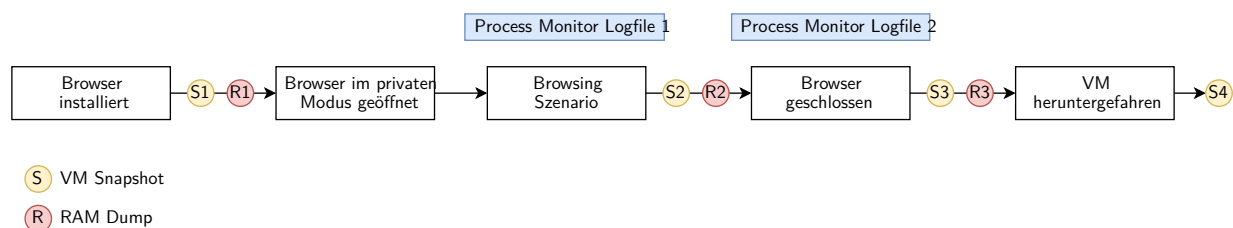


Abbildung 4.8: Datensammlung Zeitpunkte

erste RAM-Dump sowie der erste VM-Snapshot erstellt. Diese Speicherabbilder dienen als Benchmark für die Analyse, da in diesen Speicherabbildern kein PB Artefakt gefunden werden darf.

Nachdem der Private Modus im Browser geöffnet wird, bevor das Browsing Szenario beginnt wird die Aufnahme des ersten Process Monitor Logfiles gestartet. Die Aufzeichnung beginnt erst zu diesem

Zeitpunkt, da beim erstmaligem Öffnen der Browser einige Dateien initial angelegt werden. Um ausschließlich Schreiboperationen aufzuzeichnen, die auf das private Browsing zurückzuführen sind, wird die Aufzeichnung erst nach dem erstmaligen Öffnen des Browsers im privaten Modus gestartet.

Nach Durchführung des Browsing-Szenarios, während der Browser noch geöffnet ist, wird die Aufnahme des ersten Process Monitor Logfiles beendet. Weiterhin wird ein zweiter RAM-Dump sowie VM-Snapshot erstellt. Anschließend wird das eine zweite Process Monitor Aufzeichnung gestartet. Somit enthält das erste Logfile zu diesem Zeitpunkt die Prozessaktivitäten während des Browsing Szenarios.

Nachdem der Browser geschlossen wurde, wird die Aufzeichnung des zweiten Process Monitor Logfiles beendet. Zusätzlich wird ein dritter RAM-Dump sowie VM-Snapshot erstellt. Somit enthält das zweite Logfile wird alle Prozessaktivitäten vom Schließen der Browser.

Nach Herunterfahren der VM wird ein vierter VM-Snapshot erstellt, der für die für Post-Mortem Analyse relevant ist.

**Sonderfälle** Dieses Vorgehen zur Datensammlung wird bei allen Browsern durchgeführt. Einzig der Tor-Browser weicht davon ab. Wie in Kapitel X beschrieben, besitzt der Tor-Browser die Funktion der "Neuen Identität", eine Funktion um bestimmte Nutzerdaten zu löschen und zurückzusetzen. Um diese Funktion in das Browsing Szenario aufzunehmen, werden beim Tor-Browser zusätzlich Daten vor und nach der Erstellung einer "Neuen Identität" gesammelt. Wie in Abbildung X (TODO!) dargestellt, umfasst dies einen zusätzlichen RAM-Dump sowie VM-Snapshot und ein weiteres Process Monitor Logfile.

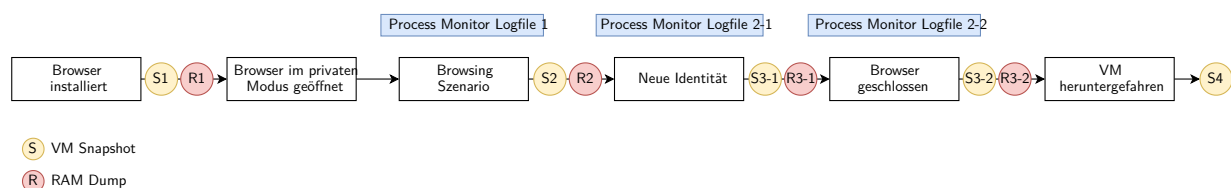


Abbildung 4.9: Datensammlung Zeitpunkte Tor

Bei Durchführung des Browsing-Szenarios für den Firefox-Browser wurde nach erstmaligem Öffnen des Browsers automatisch die Webseite <https://www.mozilla.org/de/privacy/firefox/> geöffnet. Zu diesem Zeitpunkt befand sich Firefox nicht im privaten Modus. Bei der automatisch geöffneten Seite handelt es sich um Hinweise zum Datenschutz beim Firefox- Browser.

## 4.3 Datenanalyse

Nachdem Daten in Form von Process Monitor Logfiles und Festplatten- sowie RAM-Speicherabbildern gesammelt: Daten analysieren.

Analyse heißt bei Browser Forensik: Suchen nach PB Artefakten aus Browsing Szenario in gesammelten Daten = Suchen nach Strings in Tabelle X aus Kapitel X (TODO!), der einem konkreten Schritt im Browsing Szenario entspricht.

Wichtig dabei: gefundenes Artefakt muss eindeutig Browser zugeordnet werden können. Oft kritisiert: Viele Autoren (TODO: Quellen) verlassen sich beispielsweise bei der Analyse der RAM-Dumps auf eine einfache Stringsuche in einem Hexadezimal-Editor. Sagt jedoch nichts darüber aus, ob gefundener String tatsächlich auf privates Browsen zurückzuführen ist. \*\*\* TODO: String in Editor Beispiel \*\*\*

Mit dieser Anforderung lassen sich die gesammelten Daten des Versuchs in drei Kategorien aufteilen:  
> Common Locations > Uncommon Locations > Registry

### 4.3.1 Common Locations

\*\*\* TODO: Wichtig: NUR FESTPLATTE, NICHT RAM \*\*\*

Die sogenannten "Common Locations", (dt. "gängige Speicherorte") beziehen im Zusammenhang der Browserforensik auf die standardmäßigen Verzeichnisse eines Browsers. Dazu zählen beispielsweise Ordner von Browsern zur Verwaltung von Nutzerdaten.

Untersucht werden Common Locations mittels "Whitebox-Analyse"[2] Der forensische Analyst besitzt dabei über umfassende Kenntnisse über den Browser und hat vollständigen Zugriff auf das untersuchte System. Der Fokus liegt darauf, das System vollständig zu verstehen und alle relevanten Beweise zu sammeln. Dazu werden zuerst die gängigen Browser-Speicherorte identifiziert. Im Falle dieses Versuchs werden die Browser Speicherorte über die Schreiboperationen der Process Monitor Logfiles identifiziert. Anschließend wird für jede Datei in den Speicherorten geprüft, ob PB Artefakte enthalten sind. Dazu sind zwei Schritte notwendig:

1. Dateiextraktion: Extraktion der Datei aus dem Speicherabbild. Wenn die Datei nicht mehr vorhanden ist, werden dazu ggf. Tools zur Dateiwiederherstellung benötigt.
2. Dateianalyse: Um zu überprüfen ob die Datei PB Artefakte enthält, werden ggf. Tools für spezielle Dateiformate benötigt, beispielsweise Dekomprimierungstools.

Die Untersuchung der Common Locations wird im Zusammenhang der Browser-Forensik auch "Triage" genannt [10]. Wenn für den vorliegenden Browser die gängigen Speicherorte bekannt sind, kann im ersten Analyseschritt gezielt nach Dateien gesucht werden, die aus Erfahrungswerte PB Artefakte beinhalten.

### Process Monitor Logfiles

**Identifikation der Common Locations** Um die gängigen Browserpfade und -dateien zu identifizieren, werden die in den Process Monitor Logfiles aufgezeichneten Schreibaktivitäten der Browserprozesse ausgewertet.

Dazu wird jede Process Logfile mit dem Process Explorer eingelesen. Anschließend werden die Aktivitäten gefiltert. Wie in Abbildung X dargestellt werden dazu ausschließlich die Option "File System Activity" ausgewählt. Anschließend wird als Prozessname der Browserprozess gesetzt:

**Firefox** firefox.exe

**Tor-Browser** firefox.exe und tor.exe



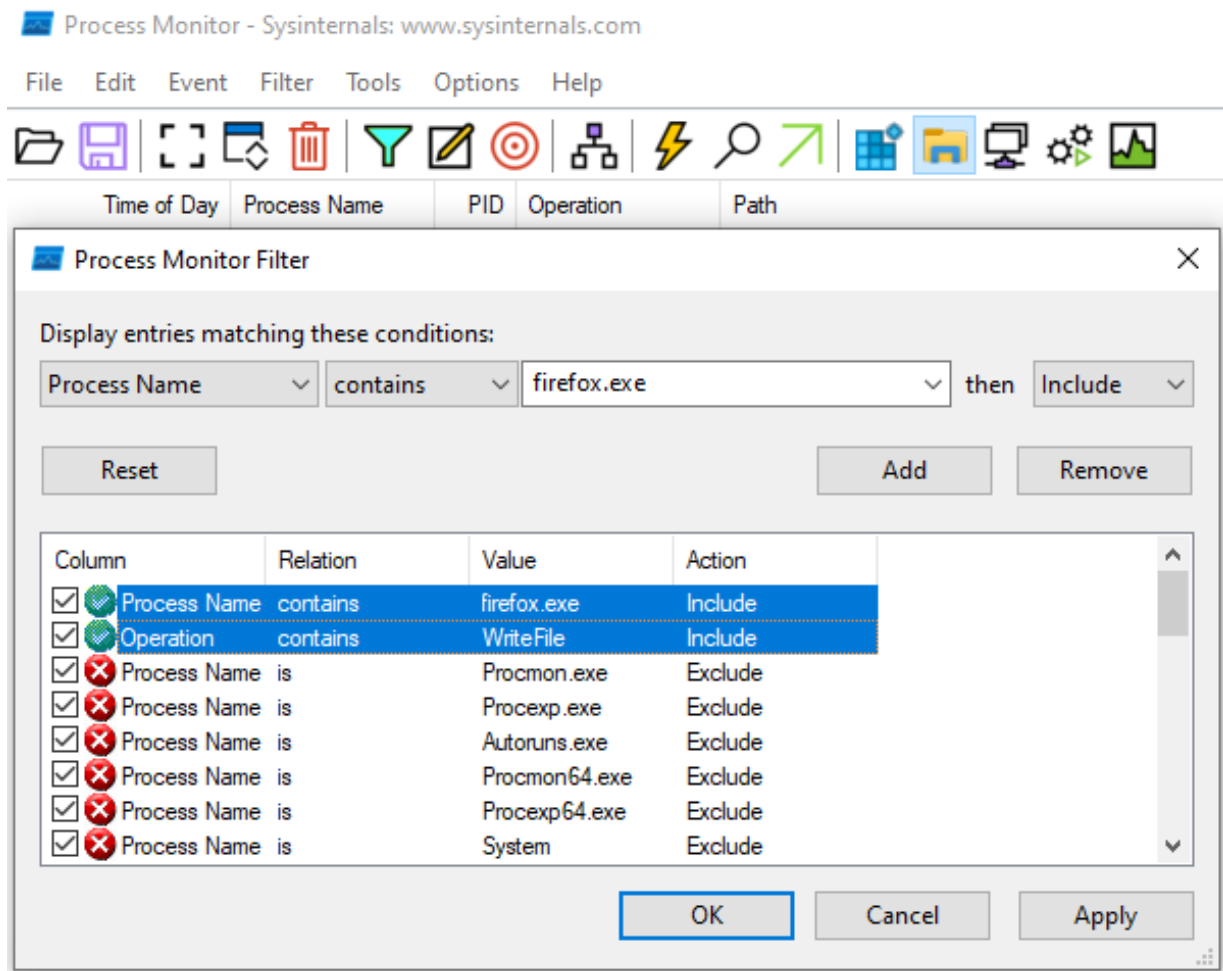


Abbildung 4.10: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

**Chrome** chrome.exe

**Brave** brave.exe

Weiterhin wird als Prozessoperation "WriteFile" gesetzt, um ausschließlich Schreibaktivitäten zu filtern. Hintergrund ist, dass PB Artefakte nur über "WriteFile" Operationen entstehen können, nicht über beispielsweise Löschoperationen.

Anschließend wird die gefilterte Logfile als CSV exportiert, um sie dann in Excel zu öffnen. In Excel werden nun folgende für den Versuch irrelevante Spalten gelöscht:

- Time of Day: Die zeitliche Reihenfolge innerhalb einer Logfile wird nicht berücksichtigt
- Process Name: Im Process Monitor wurde bereits nach Prozessnamen gefiltert, wodurch alle Prozesse den gleichen Prozessnamen haben.
- Operation: Im Process Monitor bereits nach der Operation „WriteFile“ gefiltert, wodurch alle Prozesse die gleiche Operation haben

- Result
- Detail

Danach werden gleiche mehrfache, gleiche Operationen gelöscht. Abschließend werden alle geschriebenen Dateien nach nach browserspezifischen Speicherorten gruppiert.

**Prüfung auf PB Artefakte** Nachdem die geschriebenen Browserdateien identifiziert und kategorisiert wurden, wird für jede Datei geprüft, ob PB Artefakte enthalten sind. Die zur Dateieextraktion sowie Dateianalyse notwendigen Schritte sind in Abbildung X (TODO!) dargestellt.

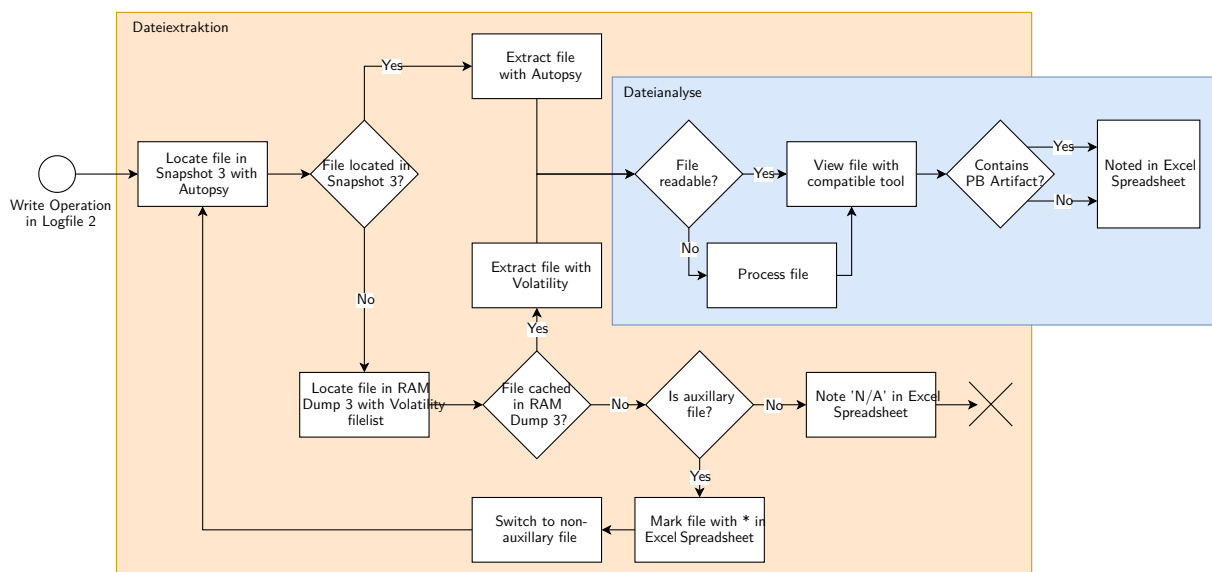


Abbildung 4.11: TODO: Process Monitor Write Operation to Excel Spreadsheet

Zur Dateieextraktion sind werden folgende Schritte durchgeführt.

1. In Autopsy prüfen, ob Datei in Festplatten-Image des entsprechenden VM-Snapshots vorhanden ist. Welcher Snapshot bei welchem Logfile untersucht wird ergibt sich aus dem Browsing Szenario in Kapitel X und ist in Tabelle X dargestellt. Hier werden nur VM-Snapshots der Live-Forensik untersucht. Der vierte VM-Snapshot zählt zur Post-Mortem-Forensik und ist erst in späteren Analyseschritten relevant (siehe Kapitel X, Y, Z TODO!)

	Logfile	Untersuchter Snapshot
Firefox, Chrome,	Logfile 1	Snapshot 2
	Logfile 2	Snapshot 3
Brave Tor	Logfile 1	Snapshot 2
	Logfile 2-1	Snapshot 3-1
	Logfile 2-2	Snapshot 3-2

2. Wenn ja, Datei mit Autopsy extrahieren.

3. Wenn nein, prüfen, ob Datei in Arbeitsspeicher vorhanden. Dazu wird die Ausgabe des Volatility Plugins "filescan" angewendet auf den entsprechenden RAM-Dump überprüft. Dazu wird der Befehl `vol.py -f ram_dump.img windows.filescan > filescan.txt` ausgeführt. Die Ausgabe von Filescan enthält eine Liste von Dateinamen, die im Speicher gefunden wurden. Welcher RAM-Dump bei welchem Logfile untersucht wird, ergibt sich aus dem Browsing Szenario in Kapitel X und ist in Tabelle X dargestellt. Wenn eine Datei in der Ausgabe gefunden wurde,

	Logfile	Untersuchter RAM-Dump
Firefox, Chrome,	Logfile 1	RAM-Dump 2
	Logfile 2	RAM-Dump 3
Brave Tor	Logfile 1	RAM-Dump 2
	Logfile 2-1	RAM-Dump 3-1
	Logfile 2-2	RAM-Dump 3-2

wird sie mithilfe des Volatility "dumpfiles" Plugins anhand der virtuellen Dateispeicheradresse extrahiert. Diese Abhängigkeiten zwischen den beiden Plugins ist in Abbildung X dargestellt.

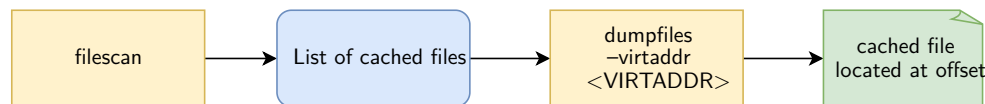


Abbildung 4.12: Datensammlung Zeitpunkte Tor

4. Wenn Datei auch nicht im RAM gecacht ist: Prüfen, ob es um eine temporäre Datei (meistens Endung ".tmp") handelt.
5. Wenn es sich um eine temporäre Datei handelt: Wird die entsprechende Nicht-temporäre gewählt und und erneute bei Schritt 1 begonnen. Wenn beispielsweise die Datei `some-file.json.tmp` nicht existiert wird geprüft prüfen, ob die Datei `some-file.json` existiert.
6. Wenn es sich um keine temporäre Datei handelt: Datei als "nicht wiederherstellbar" markieren

Nachdem eine Datei extrahiert werden konnte, werden folgende Schritte zur Dateianalyse durchgeführt.

8. Datei mit entsprechendem Tool untersuchen
9. Wenn Datei nicht lesbar ist: Datei mit zusätzlichem Tool vorverarbeiten. Beispielsweise werden komprimierte Dateien dekomprimiert.
10. In Excel-Tabelle markieren, ob die Datei private Browsing Artefakte enthält. Dabei gibt es drei Zustände:
  - leere Datei
  - neuer (nicht-leerer) Inhalt
  - gleichbleibender Inhalt

## SQLite-Datenbanken

Eine besondere Rolle unter den Common Locations bei Browsern nehmen SQLite Datenbanken ein. SQLite ist eine relationale Datenbank-Engine. Sie ermöglicht das Speichern und Verwalten von Daten in einer einzigen Datei, ohne einen separaten Datenbankserver zu benötigen. Diese Datenbanken speichern bei Browsern Nutzerinformationen wie Lesezeichen, Browserverlauf, Caches, Cookies und Erweiterungsdaten. Sie bieten einerseits Benutzern eine personalisierte Browsererfahrung und sind andererseits von forensischer Bedeutung, um Benutzeraktivitäten zu analysieren.

Aus diesem Grund werden für jeden Browser die SQLite Datenbanken aller Snapshots miteinander verglichen. Wie in Abbildung X dargestellt erfolgt die Dateieextraktion analog zur Vorgehensweise bei

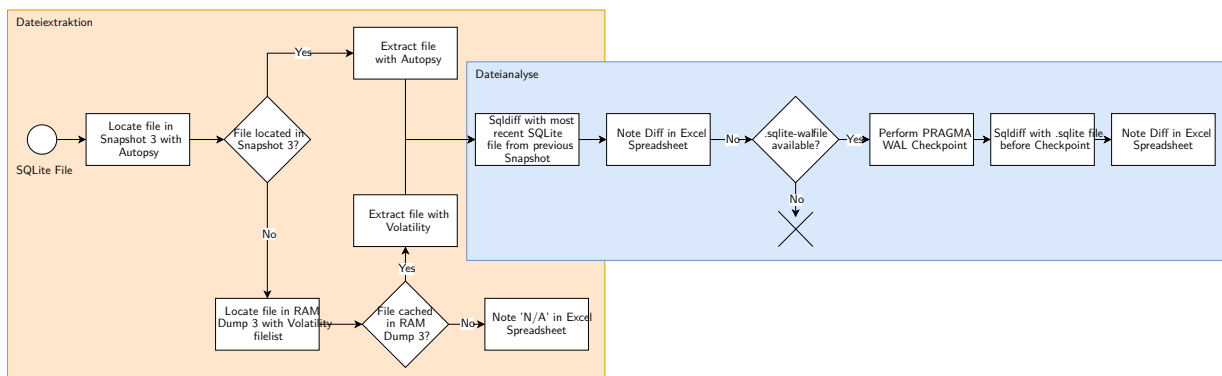


Abbildung 4.13: TODO: Process Monitor Write Operation to Excel Spreadsheet

den Schreiboperationen der Process Monitor Logfiles.

Um die SQLite Datenbanken zu analysieren wird jede Datenbank mit der gleichen Datenbank aus dem vorherigem Snapshot verglichen. Dazu wird das Befehlszeilentool `sqldiff.exe` verwendet, das über den Befehl `sqldiff.exe database1.sqlite database2.sqlite` Inhaltsunterschiede zwischen SQLite-Datenbanken anzeigt. Diese Unterschiede werden für jede Datei in jedem Snapshot untersucht und in einer Excel Tabelle festgehalten.

Zu jeder SQLite Datenbank gibt es normalerweise eine `.sqlite-wal` Datei. Dabei handelt es sich um den sogenannten "Write-Ahead Log", kurz WAL. Dort werden Datenbankänderungen vorübergehend protokollieren, bevor sie dauerhaft in die Hauptdatenbankdatei geschrieben werden. Um potentielle PB Artefakte im WAL zu berücksichtigen, werden die Inhalte des WAL in die SQLite Datei geschrieben. Dazu wird die `.sqlite` Datei mit dem Befehl `sqlite3 <Datenbank>.sqlite` über die `sqlite3` Kommandozeile geöffnet, um anschließend den WAL in die Hauptdatenbank zu übertragen: `sqlite3> PRAGMA wal_checkpoint;`

### 4.3.2 Uncommon Locations

Ungewöhnliche Speicherorte beziehen sich auf Verzeichnisse, die nicht zu den gängigen Speicherorten gehören. Bei Festplatten-Images handelt es sich dabei meist um Dateien des Betriebssystems oder andere Festplattenbereiche, wie beispielsweise unallokierte Speicherbereiche. Im Unterschied zu den "Common Locations" zählt zu den "Uncommon Locations" der Arbeitsspeicher.

Uncommon Locations werden mithilfe der "Blackbox-Analyse" untersucht [2]: Dies umfasst die Durchsicht des Beweismaterials ohne Vorwissen über das Browserverhalten sowie ohne Vorverarbeitung der Dateien. Im Kontext der Browser Forensik werden dazu Stringsuchen nach PB Artefakten über die gesamten Speicherabbilder durchgeführt. Somit wird die Suchrichtung der "Common Locations" umgekehrt: Ausgangspunkt ist nicht eine Datei, die nach allen PB Artefakten durchsucht wird. Stattdessen wird das gesamte Speicherabbild nach einem konkreten PB Artefakt durchsucht.

Dies ist nur durch Unterstützung mit Forensik-Tools möglich. Somit wird bei der Analyse der Uncommon Locations in die Vollständigkeit der Tools vertraut.

### Analyse mit Autopsy

Bei den Common Locations wurde Autopsy ausschließlich zur Dateixtraktion genutzt. Bei den Uncommon Locations wird Autopsy als forensisches Werkzeug zur Analyse der Festplatten-Images verwendet. Dazu wird eine Stichwortsuche mit den in Tabelle X definierten PB Artefakten über das gesamte Festplatten-Image durchgeführt.

Autopsy bietet dazu eine Suchfunktion an. Wie in Abbildung X dargestellt, kann damit nach exakten Strings, Teilstrings oder regulären Ausdrücken in Dateinamen und Dateiinhalten gesucht werden kann.

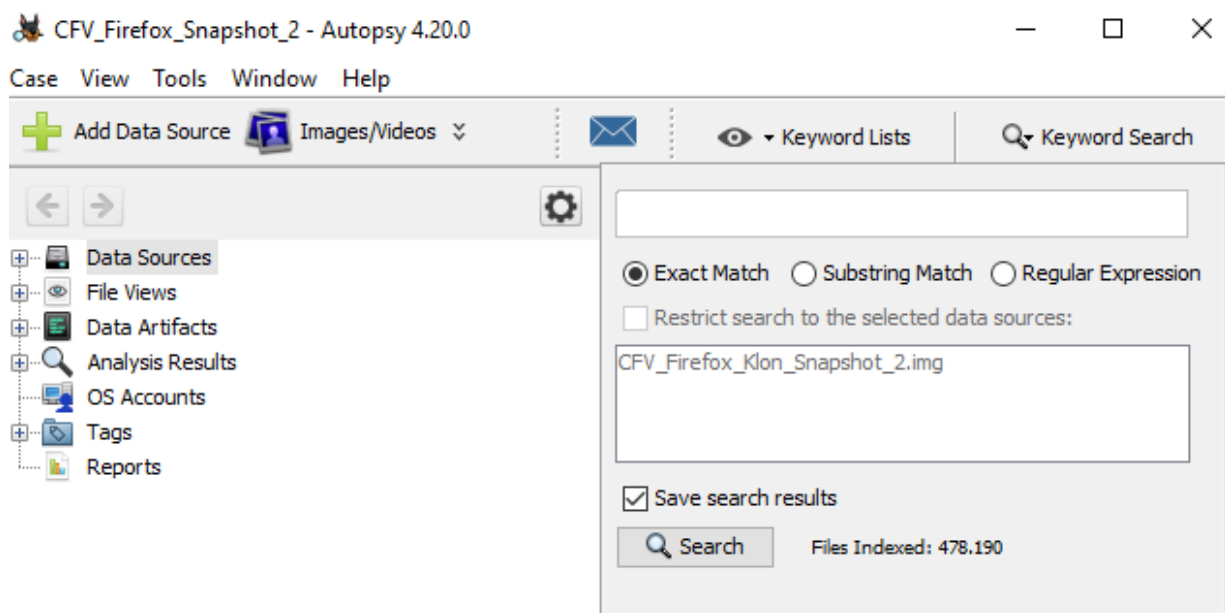


Abbildung 4.14: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

Zusätzlich kategorisiert Autopsy beim Einlesen eines Festplatten-Images automatisch die Dateien. Für jede Kategorie werden die kategorisierten Dateien nach PB Artefakten untersucht. Für diesen Versuch sind folgende Dateikategorien von Interesse:

- Web Bookmarks
- Web Cookies

- Web History
- Web Categories

Sowohl die Stichwortsuche als auch die Analyse der kategorisierten Dateien wird für jedes Festplatten-Image durchgeführt.

### Analyse mit Volatility

Bei der Analyse des Arbeitsspeichers als Uncommon Location ist es kritisch, dass ein gefundener String eindeutig einem Browserprozess zugeordnet werden kann.

In der Literatur wird der Arbeitsspeicher oft analysiert, indem der vollständige RAM-Dump als Binärdatei in einem Hexadezimaeditor wie HxD geöffnet wird, um danach eine Stringsuche durchzuführen. [13, 14, 21] Ein in der Binärdatei gefundener String ist jedoch kein Indiz, dass das Artefakt tatsächlich im Zusammenhang mit dem Browsing Szenario steht. Wie in Abbildung X und Y gezeigt, wird ein String, der in einer Textdatei auf dem Desktop gespeichert ist ebenfalls im Hexadezimaeditor HxD angezeigt, obwohl kein Browsing Szenario durchgeführt wurde.

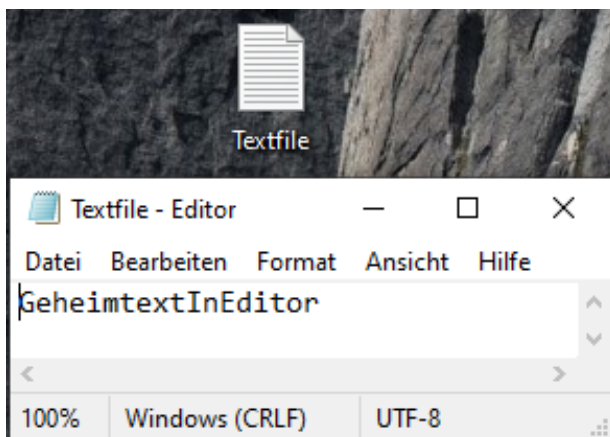


Abbildung 4.15: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

Aus diesem Grund wird das forensische Tool "Volatility" zur Analyse des Arbeitsspeichers verwendet. Bei den Common Locations wurde Volatility ausschließlich zur Dateixtraktion genutzt. Um einen im RAM gefundenen String einem Browserprozess zuordnen zu können, wird das Volatility Plugin "Yarascan" verwendet.

YARA ist ein flexibles regelbasiertes Tool, das verwendet werden kann, um nach bestimmten Mustern, Signaturen oder Verhaltensmerkmalen im Arbeitsspeicher zu suchen. Dazu werden sogenannte "YARA-Regeln" verwendet. Dabei handelt es sich um eine domänenspezifische Skriptsprache, mit Hilfe dieser zu suchende Strings und Muster definiert werden. Die für diesen Versuch verwendeten Yara-Regeln sind einfache String-Pattern, die jeden Schritt des Browsing-Szenarios abdecken. Neben den in Tabelle X in Kapitel X definierten Strings der PB Artefakte, wurde eine zusätzliche "HTML-Regel" eingeführt [22]. Diese Regel sucht nach HTML-Fragmenten, die eindeutig zu einer besuchten Seite des Browsing-Szenarios zuzuordnen sind. Die für diesen Versuch verwendeten Yara-Regeln

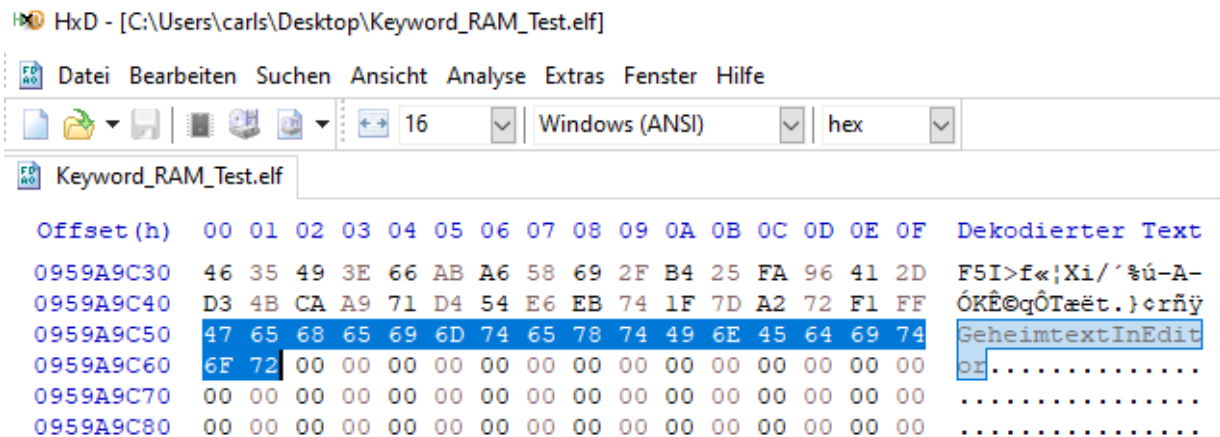


Abbildung 4.16: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

sind im Anhang X (TODO!) gezeigt. Um den RAM-Dump nach den Yara-Regeln zu durchsuchen wird folgender Befehl ausgeführt: `vol.py -f ram_dump.img windows.vadyarascan -yara-file yara_rules.yara > yarascan.txt` Nachdem der RAM-Dump nach den Regeln durchsucht wurde, liefert Yarascan eine Ausgabe mit allen gefundenen Strings. Für jeden Treffer liefert Yarascan die PID des Prozesses, in dem der String gefunden wurde sowie virtuelle Adresse des gefundenen Strings.

Wie in Abbildung X dargestellt, bildet Yarascan die Grundlage der RAM-Analyse. Davon ausgehend werden mit dem Plugin "pslist" der Prozessname zur PID des Prozesses ermittelt, in dem der String gefunden wurde. Dazu wird folgender Befehl ausgeführt: `vol.py -f ram_dump.img windows.pslist > pslist.txt`

Oft ist für bei einem gefundenen String von Interesse, ob in den Speicheradressen vor und nach dem Treffer weitere Zusammenhänge erkennbar sind. Dazu wird mithilfe des Plugins "memmap" zunächst die Abbildung der virtuellen Speicheradressen eines Prozesses auf den Byte-Offset in der Speicherseite des Prozesses: `vol.py -f ram_dump.img windows.memmap -pid <PID> > memmap.txt` Diese Seite kann mithilfe des "-dump"Flags extrahiert werden: `vol.py -f ram_dump.img -o dump_dir windows.memmap -pid <PID> -dump` Wenn die extrahierte Seite mit einem Hexadezimaleditor wie HxD geöffnet wird, kann der String-Treffer anhand des ermittelten Byte-Offsets gefunden werden.

### 4.3.3 Registry

Die letzte Kategorie analysierter Daten umfasst die Artefakte der Registry. Diese zählen sowohl zu den Common als auch Uncommon Locations und werden deshalb eigene Kategorie aufgeführt.

**Common Locations** \*\*\* TODO: Common location: Shellactivities Key \*\*\* existiert nicht mehr -> Nicht mehr vorhanden in aktueller Version (Verweis auf E-Mail)

Als Teil der Common Locations werden die Registry-Aktivitäten in den Process Monitor Logfiles analysiert. Dazu wird jede Logfile in Process Monitor eingelesen und wie in Abbildung X gezeigt

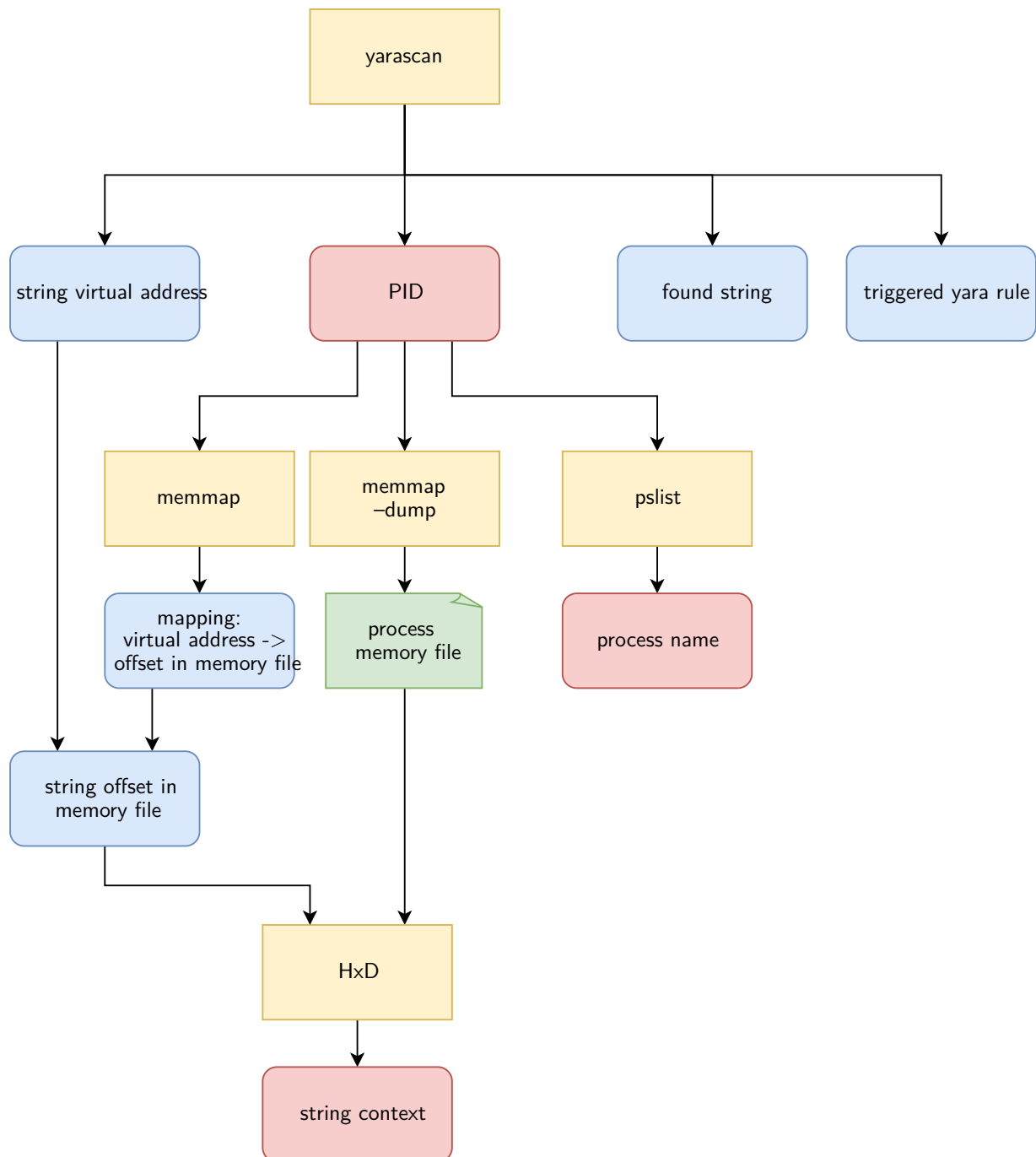


Abbildung 4.17: TODO: Process Monitor Write Operation to Excel Spreadsheet

gefiltert. Zunächst wird ausschließlich die Option "Registry Activity" ausgewählt. Anschließend wird nach Browser-Prozessnamen analog zu den Schreiboperationen in Kapitel X gefiltert. Das gefilterte Logfile wird als CSV Datei exportiert und in Excel weiter verarbeitet. Zunächst werden gleiche Schreiboperationen gelöscht. Anschließend werden die geschriebenen Registry Keys browserspezifisch



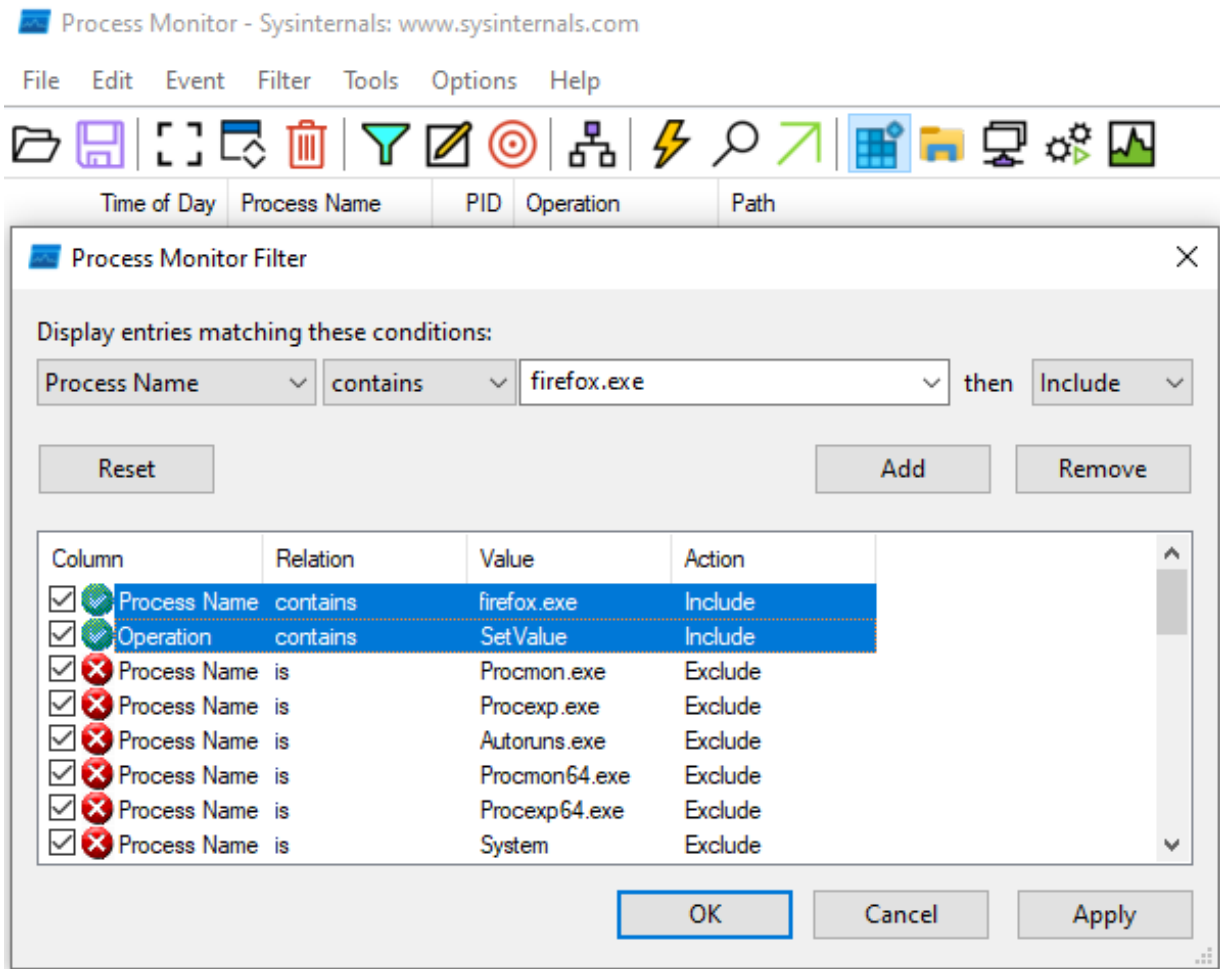


Abbildung 4.18: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

gruppiert.

**Uncommon Locations** Als Uncommon Location werden alle Registry Hives in jedem Festplatten-Image mit dem "Registry Explorer untersucht. Jedes Hive hat eine bestimmte Funktion, wie die Speicherung von Systemeinstellungen (System-Hives) oder individuellen Benutzerkonfigurationen (User-Hives). Diese in Tabelle X dargestellten Hives werden von Windows beim Start geladen und dienen als Quelle für Einstellungen und Informationen, die von verschiedenen Systemkomponenten und Anwendungen genutzt werden. Zur Analyse wird jeder Hive aus einem VM-Snapshot extrahiert und in eine Registry Explorer Sitzung geladen, anschließend wird eine Stringsuche nach PB Artefakten in allen geladenen Hives gleichzeitig durchgeführt.

System-Hives (C:\\Windows\\System32\\Config)	
Dateiname	Inhalt
DEFAULT	Standardkonfigurationseinstellungen für neue Benutzerprofile.
SAM	Sicherheitskontensdaten, einschließlich der Benutzerkonten und deren Kennwörter.
SECURITY	Sicherheitsinformationen für die Zugriffssteuerung und Authentifizierung.
SOFTWARE	Konfigurationsdaten für installierte Software und Anwendungen.
SYSTEM	Systemkonfigurationseinstellungen und Gerätetreiberinformationen.

User-Hives (C:\\Users\\<username>)	
Dateiname	Inhalt
NTUSER.DAT	Individuelle Einstellungen und Konfigurationen für den angemeldeten Benutzer
USRCLASS.DAT	Dateizuordnungen und Registrierungseinstellungen für den angemeldeten Benutzer

## 5 Ergebnisse

\*\*\* TODO \*\*\*

### 5.1 Firefox

Im nachfolgenden Abschnitt werden die Ergebnisse der Datenanalyse für den Webbrowser Firefox detailliert beschrieben. Die Analyse ist in drei Hauptkategorien unterteilt: Common Locations, Uncommon Locations und Registry.

#### Common Locations

Zunächst werden die standardmäßigen Speicherorte für Browserartefakte nach potentiellen privaten Browsing Artefakten untersucht. Diese Common Locations beziehen sich ausschließlich auf Dateien, die auf die Festplatte geschrieben werden. In diesem Versuch wird gemäß Methodik in Kapitel X (TODO!) zwischen Schreiboperationen aus den Process Monitor Logfiles und SQLite Datenbanken zur Verwaltung von Nutzerdaten unterschieden.

#### Process Monitor WriteFile Operations

Gemäß Versuchsdurchführung in Abbildung X (TODO!) wurden für Firefox mit dem Process Monitor Tool zwei Logfiles erstellt. Diese Dateien enthalten alle aufgezeichneten Prozessaktivitäten während und nach dem Browsing Szenario. Zunächst werden die beiden Logfiles gemäß Methodik in Kapitel X (TODO!) in Excel aufbereitet. Im Anhang X (TODO!) ist dazu eine Tabelle mit allen in den gefilterten Logfiles identifizierten Dateien aufgeführt. Dabei wurde für jede Datei vermerkt ob und wie sie wiederherstellbar war, mit welchem Tool die Datei analysiert wurde und ob PB Artefakte enthalten sind.

Abbildung X (TODO!) zeigt diese Tabelle in reduzierter Darstellung. Dazu wurden ausschließlich wiederherstellbare Dateien aufgeführt. Die Dateien wurden in die fünf Kategorien "Cache", "datareporting", "SSessionstore-Backup" und "SSonstige" Dateien eingeordnet. Für jede Datei wurde vermerkt, ob in der entsprechenden Logfile PB Artefakte geschrieben wurden. Dies trifft für keine der identifizierten Dateien zu.

Bei detaillierter Untersuchung der Dateien, können zwei Pfade identifiziert werden, in die Firefox während des Versuchs Dateien schreibt. Nur die Dateien in der Cache Kategorie sind im Local Pfad gespeichert.

		Logfile 1	Logfile 2
Cache	cache2\entries\037778A55E1B7E9BED3390289866D09402D6C913	Keine PB Artefakte	Keine Schreiboperation
	cache2\entries\1223A0378B8971FA4CD25EA1731C80B2B1676B42	Keine PB Artefakte	Keine Schreiboperation
	cache2\entries\250EE2BC03AFF526F1A1C3DB212A79DE3EB60D5E	Keine PB Artefakte	Keine Schreiboperation
	jumpListCache\ZKJGVJPzPe7w4w0KwEY0jw==.ico	Keine PB Artefakte	Keine Schreiboperation
	cache2\index.log	Keine Schreiboperation	Keine PB Artefakte
Datareporting	cache2\index	Keine Schreiboperation	Keine PB Artefakte
	datareporting\glean\events\pageload	Keine PB Artefakte	Keine Schreiboperation
	*datareporting\glean\db\data.safe	Keine PB Artefakte	Keine PB Artefakte
	*datareporting\archived\2023-05\1683405837882.9102466b-e465-4ecb-810f-74ae90c64c63.new-profile.jsonlz4	Keine Schreiboperation	Keine PB Artefakte
	*datareporting\archived\2023-05\1683405837905.86f4c992-6329-415b-8c29-911a2d4b7f9d.event.jsonlz4	Keine Schreiboperation	Keine PB Artefakte
SQLite	*datareporting\archived\2023-05\1683405837939.abf8b065-41a4-4e94-a044-1cead61e396a.main.jsonlz4	Keine Schreiboperation	Keine PB Artefakte
	storage\permanent\chrome\idb\3870112724rsegmnolttet-es.sqlite	Keine PB Artefakte	Keine Schreiboperation
	storage\permanent\chrome\idb\1657114595AmcateirvtiSty.sqlite	Keine PB Artefakte	Keine PB Artefakte
	places.sqlite	Keine PB Artefakte	Keine PB Artefakte
	cookies.sqlite	Keine PB Artefakte	Keine Schreiboperation
	formhistory.sqlite	Keine PB Artefakte	Keine Schreiboperation
	webappsstore.sqlite	Keine Schreiboperation	Keine PB Artefakte
	favicons.sqlite	Keine PB Artefakte	Keine Schreiboperation
	storage.sqlite	Keine PB Artefakte	Keine Schreiboperation
	*sessionstore-backups\recovery.jsonlz4	Keine PB Artefakte	Keine PB Artefakte
Sonstige	prefs-1.js	Keine PB Artefakte	Keine PB Artefakte
Dateien	*xulstore.json	Keine PB Artefakte	Keine PB Artefakte

Abbildung 5.1: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

**Local** C:\Users\<User>\AppData\Local\Mozilla\Firefox\Profiles\<Profile>.default-release\

**Roaming** C:\Users\<User>\AppData\Roaming\Mozilla\Firefox\Profiles\<Profile>.default-release\

In Tabelle X (TODO!) sind die Dateien je nach Speicherort "Local" (Hellblau) oder "Roaming" (Dunkelblau) entsprechend eingefärbt.

**Cache** Firefox verwendet den Cache, um Webseiten und deren Ressourcen temporär lokal zu speichern. Dadurch können wiederholte Anfragen an den Server vermieden und die Ladezeiten verringert werden. Die Inhalte dieser Dateien sind binär. Die Dateien im Format \cache2\entries\<ID> werden dem Cache zugeordnet und im Local Pfad gespeichert. Wie in Kapitel X beschrieben, können diese Dateien mit dem Tool MZCacheView eingelesen werden. Wie in Abbildung X gezeigt, konnten im Cache-Ordner im zweiten Snapshot drei JSON Dateien identifiziert werden. Dabei handelt es sich um Zertifikatsdateien, die von der Öne Certificate Revocation List stammen, ein Mechanismus von Firefox zur Überprüfung von Zertifikaten. In keinem der Zertifikate konnten mit HxD private Browsing Artefakte oder besuchte Seiten gefunden werden. Weiterhin befindet sich im Cache das HTML-Dokument der Firefox Datenschutzseite, welche sich beim ersten Start des Browsers automatisch öffnete. Weitere Cache Dateien konnten in keinem Snapshot gefunden werden. Die Indexdatei \cache2\index dient als Datenbank im Cache.

MZCacheView: E:\Autopsy-Cases\CFV_Firefox_Snapshot_2\Export\cache2			
File Edit View Options Help			
Filename	Content Type	URL	File Size
1683385035447.json	application/json	https://firefox.settings.services.mozilla.com/v1/buckets/security-state/collections/cert-revocations/changeset?_exp...	45,255
_expected=1682866623436&_since=%221682110623483%22.json	application/json	https://firefox.settings.services.mozilla.com/v1/buckets/security-state/collections/intermediates/changeset?_expect...	3,256
1668019772820.json	application/json	https://firefox.settings.services.mozilla.com/v1/buckets/security-state/collections/onecr!_expected=1668019772820	1,862
firefox.htm	text/html; charset=...	https://www.mozilla.org/de/privacy/firefox	16,943

Abbildung 5.2: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

Sie ermöglicht dem Firefox-Browser, schnell auf die zwischengespeicherten Ressourcen zuzugreifen und diese effizient zu verwalten. Sowohl mit HxD als auch dem Tool FirefoxCache2 konnten keine PB Artefakte identifiziert werden. \*\*\* TODO: Erst in Logfile 2 geschrieben \*\*\*

Schließlich enthält die Datei `\jumpListCache\ZKJGVJPzPe7w4w0KwEY0jw==.ico` ein 64x64 Pixel großes Mozilla Logo. Dieses Logo ist keinem Schritt aus dem Browsing Szenario zuzuordnen

**Datareporting** Dateien im Ordner `\datareporting\glean\db` sind Teil des Glean-Systems, das für die Sammlung von Telemetriedaten und deren Übermittlung an Mozilla verwendet wird. Die Datei `data.safe.bin` enthält verschlüsselte und anonyme Informationen über die Nutzung des Browsers. In HxD konnten keine PB Artefakte gefunden werden \*\*\* TODO: Vergleich Logfile 1 vs 2 \*\*\*

Dateien im Foremat `\datareporting\glean\db\<Profilname>.new-profile.jsonlz4` speichern Informationen über das Firefox-Profil, das von Glean verwendet wird. Wie in Kapitel X beschrieben, lassen sich Dateien, im proprietären *jsonlz4*-Format mit dem Tool *dejsonlz4* dekomprimieren. Die entstandene JSON Datei wird mit dem Notepad++ JSON Plugin untersucht. Dabei konnten keine PB Artefakte gefunden werden. \*\*\* TODO: Nur in Logfile 2 geschrieben \*\*\*

**Sessionstore** Die Datei `\sessionstore-backups\recovery.jsonlz4` enthält eine Sicherungskopie der vorherigen Sitzung. Sie wird erstellt, wenn der Firefox-Browser nach einem Absturz oder einem unerwarteten Beenden neu gestartet wird."Jefferson Scher entwickelte ein Online-Tool zur Analyse von *Sessionstore-Backup* Dateien. In der Sitzungswiederherstellung konnten wie in Abbildung X gezeigt lediglich die automatisch geöffnete Seite über Firefox Datenschutzhinweise identifiziert werden. \*\*\*

#### Closed Window 1

##### Tab 1

Firefox Datenschutzhinweis — Mozilla [5/6/2023, 10:24:59 PM]  
<https://www.mozilla.org/de/privacy/firefox/>

Copyright © 2020 Jefferson Scher (BSD-3-Clause License). lz4.js © 2016 Pierre Curto (MIT License; Sept. 1, 2016). FileSaver.js © 2016 Eli Grey (MIT License; v1.3.2).

Abbildung 5.3: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

TODO: Logfile 1 vs Logfile 2 \*\*\*

**Sonstige Dateien** In der Datei `prefs-1.js` werden benutzerspezifische Einstellungen und Konfigurationen für den Firefox-Browser gespeichert. Die Datei enthält Präferenzen des Benutzers in Form von JavaScript-Objekten. Es konnten mit HxD keine PB Artefakte gefunden werden. \*\*\* TODO: Vergleich Logfile 2 \*\*\*

Schließlich speichert die Datei `xulstore.json` benutzerspezifische Anpassungen und Konfigurationen für den Firefox-Browser. In der Datei konnten mit Notepad++ keine PB Artefakte gefunden werden. \*\*\* TODO: Vergleich Logfile 2 \*\*\*

## SQLite Datenbanken

Wie in Kapitel X (Methodik, TODO!) erwähnt, werden SQLite Datenbanken als Datenstrukturen für Nutzerdaten genauer untersucht. Mithilfe der Process Monitor Logfiles wurden die in Tabelle X dargestellten SQLite-Datenbanken für Firefox identifiziert:

Datenbank	Gespeicherte Daten
places.sqlite	Informationen über Lesezeichen und Verlauf. Zu jeder besuchten Webseite: URL, Seitentitel, Zeitstempel des Besuchs etc.
cookies.sqlite	Von besuchten Webseiten verwendete Cookies.
storage.sqlite	Diverse Webdaten, z. B. Indexed-Datenbanken, Offline-Cache-Daten und andere lokale Speicherinformationen.
favicons.sqlite	Enthält Favicons (kleine Symbole in der Adressleiste) um besuchte Webseiten visuell zu identifizieren.
webappsstore.sqlite	Speichert Informationen über installierte Webanwendungen im Firefox-Browser, z.B. Berechtigungen und Einstellungen.
1657114595AmcateirvtiSty.sqlite	Datenspeicher für Activity Stream, eine personalisierte Übersicht über Browser-Aktivitäten beim Öffnen eines neuen Tabs.
3870112724rsegmnoittet-es.sqlite	Datenspeicher für Remote Settings, eine zentrale Verwaltung von benutzerspezifischen Browsereinstellungen.

Jede dieser Datenbanken wurde in allen vier Snapshots miteinander verglichen. Die Dateiextraktion und Dateianalyse erfolgte analog zur Methodik in Kapitel X (TODO!). Die Ergebnisse wurden in Tabelle X (TODO!) dargestellt.

	File	Snapshot 1:	Snapshot 2: After Browsing Scenario, Browser open		Snapshot 3: After Browsing Scenario, Browser closed		Snapshot 3: Browser closed	
		Browser Installation	Vor WAL	Nach WAL	Vor WAL	Nach WAL	Vor WAL	Nach WAL
SQLite	places.sqlite	N/A	Initialisiert, Zeilen, Einträge für Seiten, geöffnete Seiten	no diff	Indizes bei vorhandenen Seiten aktualisiert	no diff	no diff	no diff
	cookies.sqlite	N/A	Leer initialisiert (Nur Spaltennamen)	leer	leer	leer	leer	leer
	storage.sqlite	N/A	Leer initialisiert (Nur Spaltennamen)	leer	leer	leer	leer	leer
	favicons.sqlite	N/A	Leer initialisiert (Nur Spaltennamen)	leer	leer	leer	leer	leer
	webappsstore.sqlite	N/A	N/A	N/A	Leer initialisiert (Nur Spaltennamen)	leer	leer	leer
	formhistory.sqlite	N/A	Leer initialisiert (Nur Spaltennamen)	leer	leer	leer	leer	leer
	1657114595AmcateirvtiSty.sqlite	N/A	Initialisiert, 1 Zeile: "origin: chrome"	no diff	Einträge (Binärdaten) eingefügt, keine PB Artefakte (HxD)	no diff	no diff	no diff
	3870112724rsegmnoittet-es.sqlite	N/A	Initialisiert, 1 Zeile: "origin: chrome"	no diff	no diff	no diff	no diff	no diff
			Leer					
			Unverändert (nicht-leer)					
			Neuer (nicht-leerer) Inhalt					

Abbildung 5.4: Comparison of found PB artifacts between RAM Dumps

Nach Browser-Installation (Snapshot 1) existierte noch keine der SQLite-Dateien.

Nach dem Browsing Szenario (Snapshot 2) wurde festgestellt, dass alle SQLite-Datenbanken initialisiert wurden, außer webappsstore.sqlite. Dabei wurden in places.sqlite die automatisch im normalen Modus geöffnete Datenschutzhinweise Seite eingetragen. In restlichen Datenbanken wurden leer initialisiert, nur die Spaltennamen wurden eingetragen. Der Inhalt aller erstellten Datenbanken blieb nach Durchführung von PRAGMA WAL Checkpoints unverändert.

Nach Schließen des Browsers (Snapshot 3) wurden in places.sqlite die Indizes bei eingetragenen Seiten aktualisiert. Die SQLite-Datenbank 1657114595AmcateirvtiSty.sqlite erhielt ein binäres Datenobjekt als Eintrag. Bei der Untersuchung mit HxD konnten keine Artefakte gefunden werden. Weiterhin wurde webappsstore.sqlite leer initialisiert. Die restlichen Daten blieben im Vergleich mit Snapshot 2 unverändert. Ebenfalls veränderte sich nicht der Inhalt nach Durchführung von PRAGMA WAL Checkpoints.

Nach herunterfahren der VM (Snapshot 4) gab es keine Änderungen in den SQLite Datenbanken, auch nach Durchführung der PRAGMA WAL Checkpoints.

Somit wurden in den SQLite Datenbanken von Firefox keine zurückverfolgbaren PB Artefakte im privaten Modus hinterlassen.

Mithilfe des Process Monitors wurde festgestellt, dass sowohl während des Browsing Szenarios (Logfile 1) als auch danach (Logfile 2) Inhalte in Dateien geschrieben wurden. Wie zusammenfassend in Abbildung X (TODO!) dargestellt, wurde mit Ausnahme der Datareporting Dateien gab es in Logfile 1 stets mehr oder genauso viele Schreiboperationen wie in Logfile 2. Keine Schreiboperation hinterließ jedoch Private Browsing Artefakte.

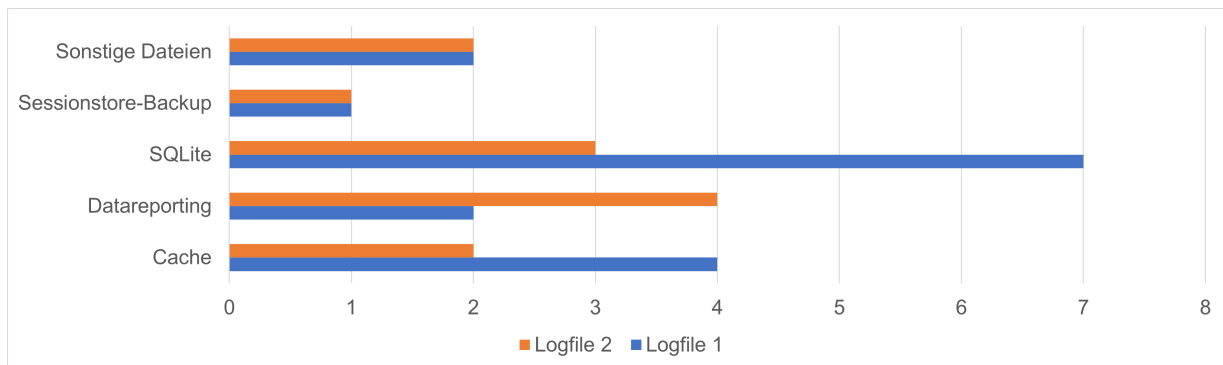


Abbildung 5.5: Comparison of found PB artifacts between RAM Dumps

## Uncommon Locations

Nachfolgend werden die Analyseergebnisse der Firefox Uncommon Locations beschrieben. Wie in Kapitel X erläutert, wird im Gegensatz zu Common Locations die Suchrichtung umgekehrt und es werden alle gesammelten Daten nach einem spezifischen PB Artefakt durchsucht. Somit benötigt ein Forensiker kein Wissen über das Browserverhalten. Stattdessen wird sich auf die Vollständigkeit der Funktionen von Forensik-Tools verlassen. Im Rahmen dieses Versuchs werden die Tools Autopsy und Volatility verwendet.

## Analyse mit Autopsy

Bei den Common Locations in Kapitel X wird Autopsy nur zur Dateieextraktion genutzt. Im Falle der Uncommon Locations dient Autopsy als forensisches Werkzeug zur Datenanalyse.

Eine Autopsy Stichwortsuche gemäß Methodik in Kapitel X (TODO!) lieferte in allen Snapshots keine Treffer. Es wurde zusätzlich das \$Carved Verzeichnis durchsucht, in dem Autopsy alle wiederhergestellten Dateien speichert.

Anschließend wurden die automatisch von Autopsy kategorisierten Dateien untersucht. Gemäß Methodik in Kapitel X wurden dazu die Dateien der Kategorien "Web Bookmarks", "Web Cookies", "Web History" sowie "Web Categories" analysiert. Beim Vergleich der Festplattenabbilder wurde festgestellt, dass ein Snapshot stets die kategorisierten Dateien des vorherigen Snapshots enthielt. Es sind innerhalb einer Kategorie nur neue Dateien dazugekommen. Somit enthält Snapshot 4 in jeder Kategorie alle Dateien der vorherigen Snapshots.

**Web Bookmarks** Bereits vor Durchführung des Browsing Szenarios enthielt Firefox im ersten Snapshot die Bing Startseite als gespeichertes Leesezeichen. In den restlichen Snapshots 2 – 4 blieb diese Kategorie unverändert.

Source Name	S	C	O	URL	Title	Date Created	Program Name	Domain	Data Source
Bing.url			19	http://go.microsoft.com/fwlink/?LinkId=255142	Bing.url	2023-04-25 16:09:28 MESZ	Internet Explorer Analyzer	microsoft.com	CFV_Firefox_Klon_Snapshot_3.img

Abbildung 5.6: Autopsy Web Bookmarks

**Web Cookies** Auch diese Kategorie enthält bereits vor Beginn des Browsing Szenarios zehn Cookie-Einträge in der Datei WebCacheV01.dat. Dabei handelt es sich um eine Datenbank des Microsoft Edge Browsers zur Speicherung von Nutzerdaten. Diese Datei verhält sich ähnlich wie die in diesem Versuch relevanten SQLite-Dateien. Die Datei enthält. Bei den Einträgen handelt es sich um Cookies für Bing und die Outlook Webseite, obwohl diese Seiten nie in Microsoft Edge geöffnet wurden. In den Snapshots 2 – 4 kamen keine weiteren Einträge in dieser Kategorie hinzu.

Source Name	S	C	O	URL	Date Accessed	Name	Value	Program Name	Domain	Data Source
WebCacheV01.dat			15	bing.com	2023-05-06 19:50:17 MESZ	SUID	M	Microsoft Edge Analyzer	bing.com	CPV_Firefox_Klon_Snapshot_3.img
WebCacheV01.dat			15	www.bing.com	2023-05-06 19:51:24 MESZ	MUIDB	31708C5FC3CF47068AFAD1CB47D0111	Microsoft Edge Analyzer	bing.com	CPV_Firefox_Klon_Snapshot_3.img
WebCacheV01.dat			15	bing.com	2023-05-06 19:50:17 MESZ	SRCHD	AF=NOFORM	Microsoft Edge Analyzer	bing.com	CPV_Firefox_Klon_Snapshot_3.img
WebCacheV01.dat			15	bing.com	2023-05-06 19:50:17 MESZ	SRCHUID	V=26GUID=B2C50ADB8B984234A9FE14DB81DCB91D&dm...	Microsoft Edge Analyzer	bing.com	CPV_Firefox_Klon_Snapshot_3.img
WebCacheV01.dat			15	bing.com	2023-05-06 19:50:17 MESZ	SRCHUSR	DOB=20230506	Microsoft Edge Analyzer	bing.com	CPV_Firefox_Klon_Snapshot_3.img
WebCacheV01.dat			15	bing.com	2023-05-06 19:50:20 MESZ	SRCH#PGUSR	SRCHLANG=de&LUT=16834026192238JPMH=dee204058J...	Microsoft Edge Analyzer	bing.com	CPV_Firefox_Klon_Snapshot_3.img
WebCacheV01.dat			15	bing.com	2023-05-06 19:50:19 MESZ	CortanaAppUID	C164AA3A4D7E127DC66AD915CFD04C	Microsoft Edge Analyzer	bing.com	CPV_Firefox_Klon_Snapshot_3.img
WebCacheV01.dat			15	bing.com	2023-05-06 19:55:22 MESZ	ANON	A=A3B5B679A14D59B0AA027635FFFFFFF	Microsoft Edge Analyzer	bing.com	CPV_Firefox_Klon_Snapshot_3.img
WebCacheV01.dat			15	live.com	2023-05-06 19:50:30 MESZ	MUID	118A534093A9626528C5404997A966B8	Microsoft Edge Analyzer	live.com	CPV_Firefox_Klon_Snapshot_3.img
WebCacheV01.dat			15	login.live.com	2023-05-06 19:51:06 MESZ	_Host-MSAUAUTHP		Microsoft Edge Analyzer	live.com	CPV_Firefox_Klon_Snapshot_3.img

Abbildung 5.7: Autopsy Web Cookies

**Web History** Diese Kategorie listet alle Dateien mit gespeichertem Suchverlauf auf. Vor Beginn des Browsing Szenarios (Snapshot 1) enthält die Kategorie ebenfalls zwei Einträge zur Outlook Webseite in der Datei WebCacheV01.dat. Nach Durchführung des Browsing Szenarios (Snapshot 2) wurde ein Eintrag in der places.sqlite Datenbank hinzugefügt. Dabei handelt es sich um die automatisch im normalen Browsingmodus geöffnete Firefox-Standardseite über Datenschutzhinweise. Dies deckt sich mit den Beobachtungen der Common Locations in Kapitel X. Darüber hinaus enthält dieser Snapshot für die Datei WebCacheV01.dat den Eintrag file:///Z:/Logfile\_1. Dabei handelt es sich um das Process Monitor Logfile, das gemäß Methodik in Kapitel X (TODO!) über den gemeinsamen VM-Ordner zum Analyse-Rechner transportiert wurde. Ergänzt wird das in Snapshot 3 durch den Eintrag file:///Z:/Logfile\_2, dem zweiten Process Monitor Logfile. In Snapshot 4 werden in dieser Kategorie keine neuen Dateien erfasst.

Source Name	S	C	O	URL	Date Accessed	Referrer URL	Title	Program Name	Domain	Data Source	Username
places.sqlite			6	https://www.mozilla.org/de/privacy/firefox/	2023-05-06 22:25:00 MESZ	https://www.mozilla.org/privacy/firefox/	Firefox Datenschutzhinweis — Mozilla	Firefox Analyzer	mozilla.org	CPV_Firefox_Klon_Snapshot_3.img	
WebCacheV01.dat			15	https://login.live.com/oauth20_desktop.srf?lc=1031	2023-05-06 19:51:06 MESZ			Microsoft Edge Analyzer	live.com	CPV_Firefox_Klon_Snapshot_3.img	Forensik
WebCacheV01.dat			15	https://login.live.com/oauth20_authorize.srf?client_id=...	2023-05-06 19:51:08 MESZ			Microsoft Edge Analyzer	live.com	CPV_Firefox_Klon_Snapshot_3.img	Forensik
WebCacheV01.dat				file:///Z:/logfile_1	2023-05-06 20:29:36 MESZ			Microsoft Edge Analyzer		CPV_Firefox_Klon_Snapshot_3.img	Forensik
WebCacheV01.dat				file:///Z:/logfile_2	2023-05-06 20:44:19 MESZ			Microsoft Edge Analyzer		CPV_Firefox_Klon_Snapshot_3.img	Forensik

Abbildung 5.8: Autopsy Web History

**Web Categories** Diese Kategorie klassifiziert im Speicherabbild gefundene Browsing Artefakte nach Inhalt. Vor Beginn des Browsing Szenarios (Snapshot 1) werden hier bereits zwei Einträge aufgelistet. Der Eintrag bing.com wird als "SSuchmaschine" klassifiziert und live.com als "Web-Email". Wie oben erwähnt, wurden beide Seiten nie aufgerufen. Es gab keine zusätzlichen Einträge in dieser Kategorie in den Snapshots 2 bis 4.

Somit wurden in allen Kategorien ausschließlich Browsing Artefakte des Edge Browsers in der Datei WebCacheV01.dat gefunden, sowie ein Eintrag in der Firefox SQLite Datenbank places.sqlite. Die eingetragene Firefox-Standardseite deckt sich mit den Ergebnissen der Common Locations in Tabelle



Source Name	△ S	C	O	Source Type	Score	...	...	Domain	Host	Name	File Path
WebCacheV01.dat			0	File	Unknown			bing.com	bing.com	Search Engine	/img_CPV_Firefox_Klon_Snapshot_3.img/vol_vol3/Users/Forensik/AppData/Local/Microsoft/Windows/WebCache/WebCacheV01.dat
WebCacheV01.dat			0	File	Unknown			live.com	login.live.com	Web Email	/img_CPV_Firefox_Klon_Snapshot_3.img/vol_vol3/Users/Forensik/AppData/Local/Microsoft/Windows/WebCache/WebCacheV01.dat

Abbildung 5.9: Autopsy Web Categories

X. Die aufgelisteten Einträge in der Datei WebCacheV01.dat sind nicht auf Schritte des Browsing Szenarios zurückzuführen. Die Einträge sind bereits im ersten Snapshot enthalten, obwohl vor Beginn des Browsing Szenarios keine Browseraktivitäten durchgeführt wurden. Weiterhin enthält diese Datei Einträge über die Process Monitor Logfiles, welche über einen gemeinsamen VM-Ordner zum Rechner transportiert wurde, auf dem die virtuelle Maschine läuft. In keiner der Kategorien konnten private Browsing Artefakte identifiziert werden.

### Analyse mit Volatility

Nachdem die Firefox Festplattenabbilder als Uncommon Location mit Autopsy untersucht wurden, werden nachfolgend die Analyseergebnisse des RAMs als Uncommon Location beschrieben. Dazu wurde eine Stringsuche im gesamten RAM nach PB Artefakten durchgeführt. Wie in Kapitel X ausführlich beschrieben muss ein gefundener String eindeutig einem Browser zugeordnet werden können.

Deshalb wurde dazu das Volatility PlugIn "Yarascan" verwendet, ein Werkzeug um nach bestimmten Mustern im RAM zu suchen. Dazu wurden die in Tabelle X aufgeführten Yara-Regeln verwendet. Wie in Kapitel Methodik (TODO!) beschrieben, wird davon ausgehend das PlugIn "pslist" verwendet, um den Prozessnamen anhand PID zu identifizieren. Die Ergebnisse dieser Stringsuche sind nachfolgend nach Kategorie geordnet.

**Yararule HTML** In keinem der Firefox RAM Dumps wurden HTML Fragmente der besuchten Seiten gefunden. Somit wird diese Yara-Regel nicht weiter betrachtet.

**Yararule Keyword** Wie in Abbildung X (TODO!) gezeigt, wurden alle Suchbegriffe "pfaffenhofen", "nanoradar", "mooserliesl sowie "mallofamerica identifiziert im zweiten RAM Dump, nach dem Browsing Szenario mit geöffnetem Browser, identifiziert. Die Artefakte befinden sich ausschließlich im zweiten RAM Dump. Die Suchbegriffe wurden größtenteils in den Speicherbereichen von Firefox-Prozessen gefunden. Nur in zwölf Fällen wurden Suchbegriffe in anderen Prozessen identifiziert. Am häufigsten wurde der Suchbegriff "pfaffenhofen" mit 1301 gefundenen Artefakten im zweiten Firefox RAM Dump gefunden. Dies ist vermutlich auf den Google Maps Kartenbereich zurückzuführen, einen visuellen Ausschnitt der Karte, welcher bei der Google-Suche erscheint und Informationen über die geografische Lage, Straßen, Sehenswürdigkeiten und andere relevante Orte in der gesuchten Stadt zeigt. In den RAM Dumps 1 und 3 konnten Artefakte zu den Suchbegriffen identifiziert werden.

**Yararule URL** Es konnten in den Arbeitsspeicherabbildern alle besuchten URLs unitree.com, mooserliesl.de, mallofamerica.com sowie donaukurier.de identifiziert werden. Dabei wurden die meisten Artefakte nach dem Browsing Szenario mit geöffnetem Browser (RAM Dump 2) gefunden. Alle besuchten URLs wurden in diesem Dump sowohl in Firefox als auch anderen Prozessen gefunden, wobei

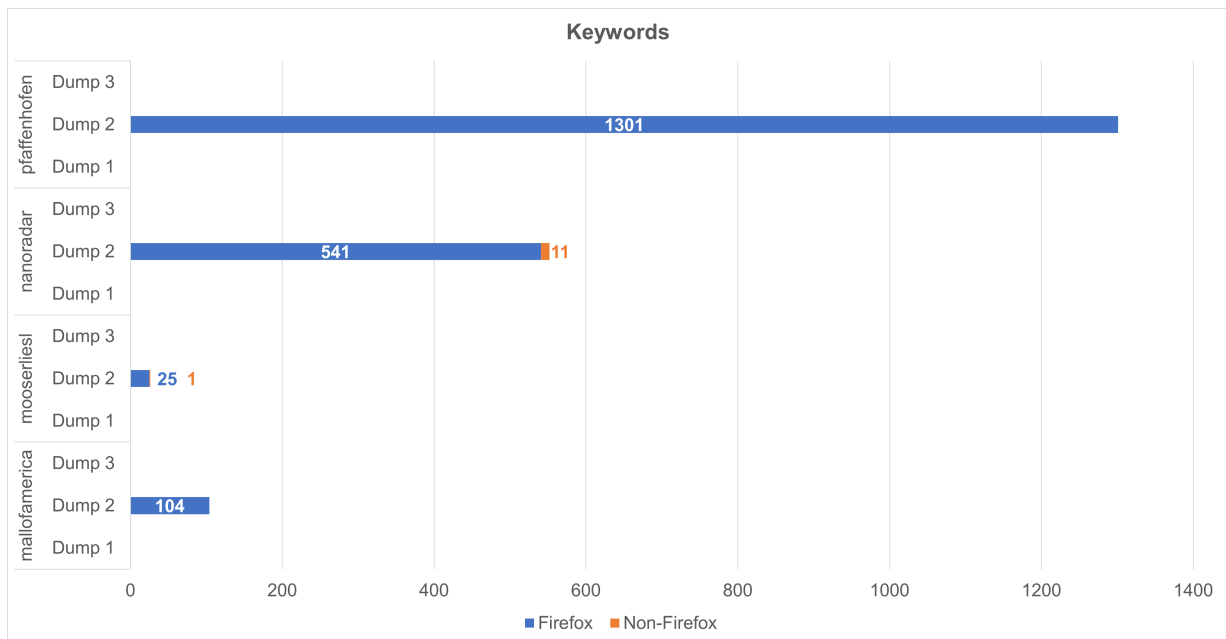


Abbildung 5.10: Keywords

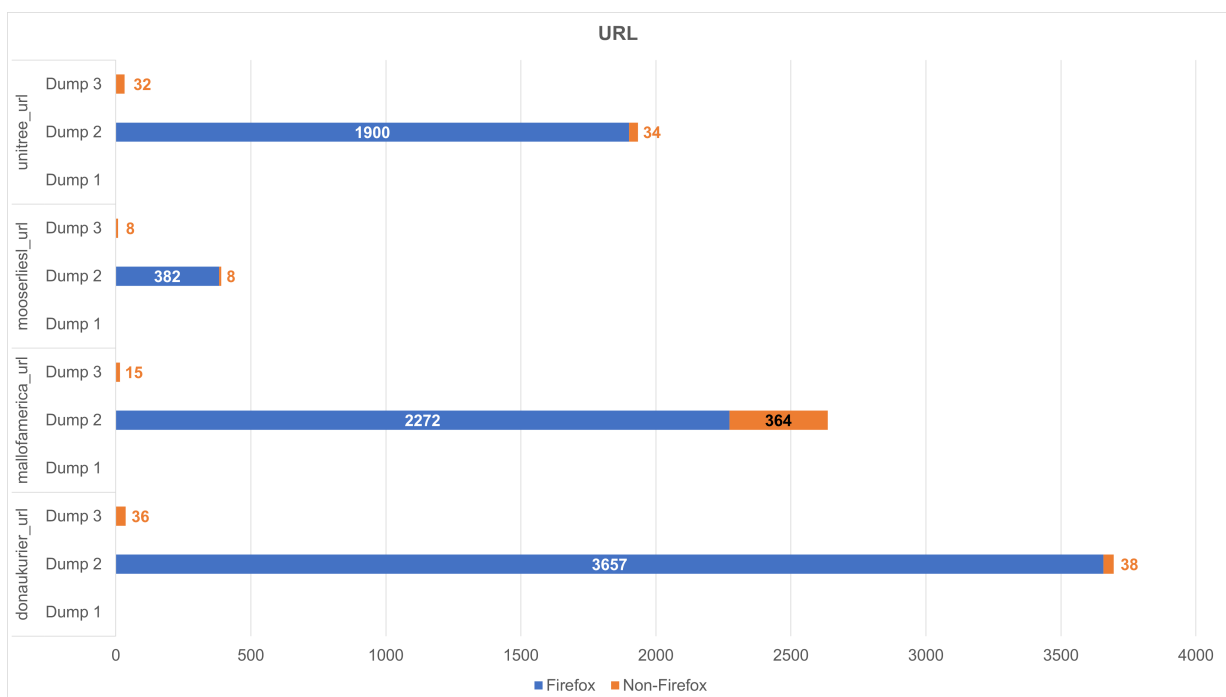


Abbildung 5.11: URL

die meisten Artefakte in Firefox Prozessen zu finden sind. Dabei wurde "mooserliesl.de" mit insgesamt 390 Artefakten am wenigsten gefunden, "donaukurier.de" mit über 3600 Artefakten am häufigsten.

Virtuelle Speicheradresse	PID	Byte-Offset in extrahierter Speicherseite
0xb9ce29180c8	7420	0x11dd40c8
0x2859f4ffd4e0	7420	0x12e234e0
0x24083b41858	8424	0x583858
0x240840e5b08	8424	0x96bb08

Bemerkenswert ist, dass URL Artefakte gefunden wurden, nachdem der Browser geschlossen wurde (RAM Dump 3). Dabei wurde kein URL Artefakte in einem Firefox Prozess gefunden. Anhand der PID 2252 wurde festgestellt, dass sich alle URL Artefakte des dritten RAM Dumps in einem "svchost.exe"-Prozess mit der gleichen PID befinden. Unter dem "SService Host"-Prozess laufen gruppierte Windows-Dienste, um Ressourcen zu sparen und die Systemleistung zu verbessern. Volatility bietet das Plugin "bvscan", mit dem alle laufenden Dienste ausgegeben werden können. Bei Anwendung auf den dritten RAM Dump wurde jedoch zu keinem Dienst eine PID angegeben, wodurch der Dienst mit den URL Artefakten nicht im RAM identifiziert werden konnte. Stattdessen wurde der dritte Snapshot aufgetaut, um im laufenden Windowsbetrieb den Dienst mithilfe des Process Explorers zu identifizieren. Wie in Abbildung X (TODO!) gezeigt, wurde anhand der PID 2252 der Dienst "DNSCache" ermittelt. Der DNSCache-Dienst unter Windows ist ein Teil des Betriebssystems, der für

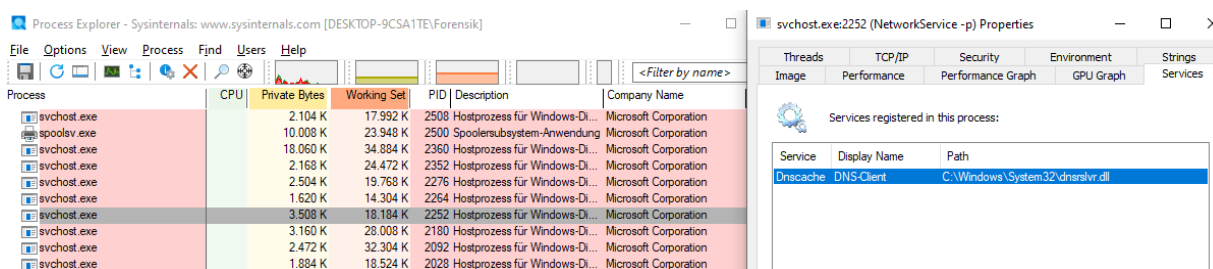


Abbildung 5.12: URL

die Übersetzung von Domainnamen in IP-Adressen verantwortlich ist. Der DNSCache-Dienst speichert DNS-Anfragen und Antworten temporär, um wiederholte DNS-Anfragen zu beschleunigen. Nach Löschen des DNSCaches mit dem Kommandozeilenbefehl `ipconfig /flushdns`, dem Schließen aller Process Monitor Instanzen sowie Beenden des DNSCaches Services wurde erneut ein RAM-Dump durchgeführt. Dort wurden keine URL Artefakte mehr gefunden.

**Yararule Mail** Es konnten alle E-Mail Artefakte des Browsing Szenarios gefunden werden. Die Artefakte befinden sich ausschließlich im zweiten Firefox RAM Dump, nach dem Browsing Szenario mit geöffnetem Browser. Unter den gefundenen Artefakten befindet sich mit zwölf Vorkommen am häufigsten die Absenderadresse "computerforensikvl@gmail.com". Dieses Artefakt wurde als einziges Mail-Artefakt in anderen Prozessen außer Firefox gefunden.

Bemerkenswert ist, dass das Passwort des Google-Accounts, mit dem die E-Mails verschickt wurden, vier mal als Klartext im RAM gefunden wurden. Das Passwort wurde in je zwei Firefox Prozessen mit den PIDs 7420 und 8424 zwei mal gefunden. Tabelle X zeigt die virtuellen Speicheradressen der Artefakte aus der Yarascan Ausgabe.

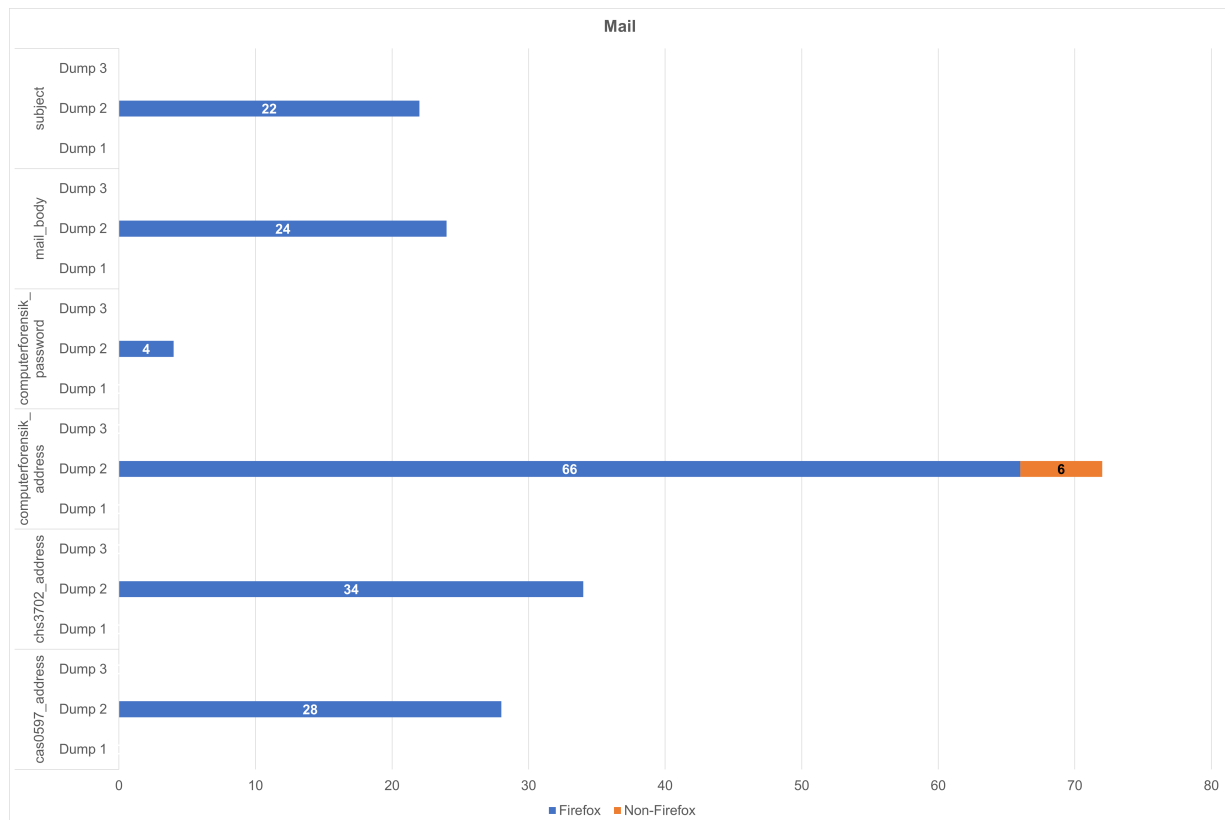


Abbildung 5.13: Mail

Zu diesen Artefakten wurde gemäß Methodik in Kapitel X der String Kontext – die Zeichen vor und nach dem gefundenen Artefakt im Speicherbereich – ermittelt. Dazu wurde gemäß Methodik in Kapitel X mithilfe des Volatility memmap-Plugins die Abbildung der virtuellen Speicheradressen auf den Byte-Offset in der extrahierten Speicherseite des Prozesses ermittelt. Wie in Abbildung X gezeigt,

```

11DD4040 58 02 00 00 08 00 00 00 67 6D 70 41 64 64 6F 6E X.....gmpAddon
11DD4050 4B 4B 4B 4B 4B 4B 4B 4B DC F9 0E 50 4B 4B 4B 4B KKKKKKKKÜä.PKKKK
11DD4060 58 02 00 00 0D 00 00 00 67 6D 70 44 6F 77 6E 6C X.....gmpDownl
11DD4070 6F 61 64 65 72 4B 4B 4B 50 C3 FB EA 4B 4B 4B 4B oaderKKKPÄüèKKKK
11DD4080 58 02 00 00 0D 00 00 00 47 4D 50 44 6F 77 6E 6C X.....GMPDownl
11DD4090 6F 61 64 65 72 4B 4B 4B D0 6F AE 8A 4B 4B 4B 4B oaderKKKBoöŠKKKK
11DD40A0 58 02 00 00 0D 00 00 00 5F 69 73 45 4D 45 45 6E X....._isEMEEEn
11DD40B0 61 62 6C 65 64 4B 4B 4B 8F 4F 0E 73 4B 4B 4B 4B abledKKK.O.sKKKK
11DD40C0 58 02 00 00 0C 00 00 00 56 6F 72 6C 65 73 75 6E X.....Vorlesun
11DD40D0 67 32 33 21 4B 4B 4B 4B F8 35 7D 48 4B 4B 4B 4B g23!KKKKø5)HKKKK
11DD40E0 58 02 00 00 0F 00 00 00 5F 69 73 41 64 64 6F 6E X....._isAddon
11DD40F0 45 6E 61 62 6C 65 64 4B 42 1D 99 C2 4B 4B 4B 4B EnabledKB.ÅKKKK
11DD4100 58 02 00 00 0F 00 00 00 6D 61 69 6C 2E 67 6F 6F X.....mail.goo
11DD4110 67 6C 65 2E 63 6F 6D 4B 44 47 D9 2D 4B 4B 4B 4B gle.comKDGÜ-KKKK
11DD4120 58 02 00 00 10 00 00 00 5F 75 70 64 61 74 65 4C X....._updateL
11DD4130 61 73 74 43 68 65 63 6B 43 1F 7D 4B 4B 4B 4B 4B astCheckC.)KKKKK
11DD4140 58 02 00 00 10 00 00 00 73 65 63 6F 6E 64 73 53 X.....secondsS

```

Abbildung 5.14: Password in memory page of PID 7420 at Byte-Offset 0x11dd40c8

sind in der Speicherseite des Prozesses mit PID 7420 konnte vor und nach dem gefundenen Passwort am Byte-Offset 0xb9ce29180c8 neben der Gmail-Url "mail.google.com" Code-Fragmente der "Gecko-

Engine zu finden. Dieser Teil des Firefox Browsers ist für das Rendering von Webinhalten verantwortlich, einschließlich HTML, CSS, JavaScript und anderen Medienformaten wie Bildern, Audio und Video. In

```

12E23470 00 00 00 00 00 00 00 00 10 02 00 00 34 00 00 00 .....4...
12E23480 00 C8 CC E5 29 02 00 00 00 00 00 00 00 00 00 00 ..Ëiä).....
12E23490 10 02 00 00 27 00 00 00 40 D3 CC E5 29 02 00 00 ....'...@öiä)...
12E234A0 00 00 00 00 00 00 00 00 10 02 00 00 2A 00 00 00 .....*.....
12E234B0 70 D3 CC E5 29 02 00 00 00 00 00 00 00 00 00 00 pÖiä).....
12E234C0 00 02 00 00 45 00 00 00 A0 47 7C 1A 55 23 00 00 ....E... G|U#..
12E234D0 F8 DE FF F4 59 28 00 00 50 02 00 00 0C 00 00 00 øÞÿôY(...P.....
12E234E0 56 6F 72 6C 65 73 75 6E 67 32 33 21 A2 1D FB FF Vorlesung23!c.üÿ
12E234F0 50 02 00 00 0A 00 00 00 69 64 65 6E 74 69 66 69 P.....identifi
12E23500 65 72 F0 B8 FA 7F 00 00 50 02 00 00 06 00 00 00 erö,ú...P.....
12E23510 50 61 73 73 77 64 F9 FF 18 96 73 E5 29 02 00 00 Passwdüÿ.-sä)...
12E23520 50 02 00 00 0E 00 00 00 73 65 73 73 69 6F 6E 72 P.....sessionr
12E23530 65 73 74 6F 72 65 00 00 10 02 00 00 2E 00 00 00 estore.....
12E23540 80 D2 CC E5 29 02 00 00 00 00 00 00 00 00 00 00 €Öiä).....
12E23550 10 02 00 00 1F 00 00 00 00 B8 CC E5 29 02 00 00 .....iä)...

```

Abbildung 5.15: Password in memory page of PID 7420 at Byte-Offset 0x12e234e0

der gleichen Datei konnte nach dem gefundenen Passwort am Byte-Offset 0x24083b41858 die Strings "Passwd sowie Sessionrestore" (siehe Common Location SSessionstore in Kapitel X) identifiziert werden. Wie in den Abbildungen X und Y (TODO!) gezeigt, können in den Byte-Offsets der gefundenen Passwörter in der Speicherseite der PID 8424 konnten kein Kontext ermittelt werden. Im Gegensatz zur Speicherseite der PID 7420 wird das Passwort dort mit 2 Bytes pro Zeichen enkodiert. Das eine Unicode-Zeichenenkodierung vermuten.

```

005837F0 02 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 .....
00583800 C0 9E EA FF 40 02 00 00 0E 00 00 00 00 00 00 00 Äzäÿ@.....
00583810 02 00 00 00 00 00 00 00 00 E5 E5 E5 E5 E5 E5 .....ääääääää
00583820 02 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 .....
00583830 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00583840 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 äääääääääääääää
00583850 02 00 00 00 1A 00 00 00 56 00 6F 00 72 00 6C 00 .....V.o.r.l.
00583860 65 00 73 00 75 00 6E 00 67 00 32 00 33 00 21 00 e.s.u.n.g.2.3.!.
00583870 00 00 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 ..ääääääääääääää
00583880 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 äääääääääääääää
00583890 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 äääääääääääääää
005838A0 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 äääääääääääääää
005838B0 08 00 00 00 D7 16 71 67 01 00 00 00 00 00 00 00 ....*.qg.....
005838C0 6F 00 6E 00 44 00 51 00 30 00 4B 00 55 00 62 00 o.n.D.Q.O.K.U.b.

```

Abbildung 5.16: Password in memory page of PID 8424 at Byte-Offset 0x583858

**Yararule Image** Das im Browsing Szenario geöffnete Donaukurier Logo wurde ausschließlich im zweiten RAM Dump in drei mal in Firefox Prozessen gefunden.

Wie in Abbildung X zusammenfassend gezeigt wurden vor dem Browsing Szenario, keine private Browsing Artefakte im ersten RAM Dump gefunden. Nach dem Browsing Szenario mit geöffnetem Browser konnten die meisten Artefakte identifiziert werden. Dabei wurden am häufigsten URL Artefakte in Firefox Prozessen gefunden. Zudem konnte hier das E-Mail Passwort im Klartext lokalisiert werden. Nach Schließen des Browsers konnten im dritten Snapshots URLs im DNSCache Windows Service gefunden werden. Nach leeren des Caches und Beenden des DNSCache Services konnten wurden keine Artefakte gefunden.

[illegible]

Abbildung 5.17: Password in memory page of PID 8424 at Byte-Offset 0x96bb08

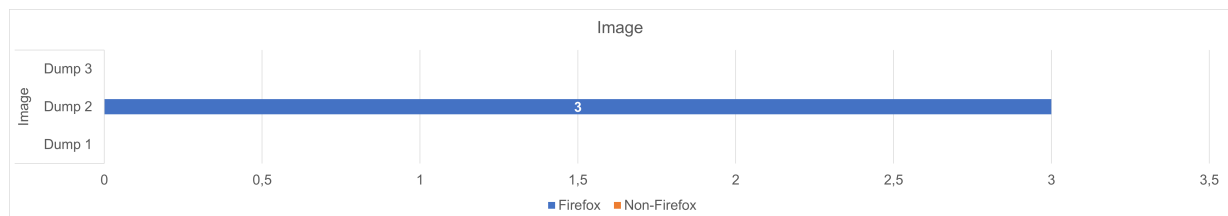


Abbildung 5.18: Image

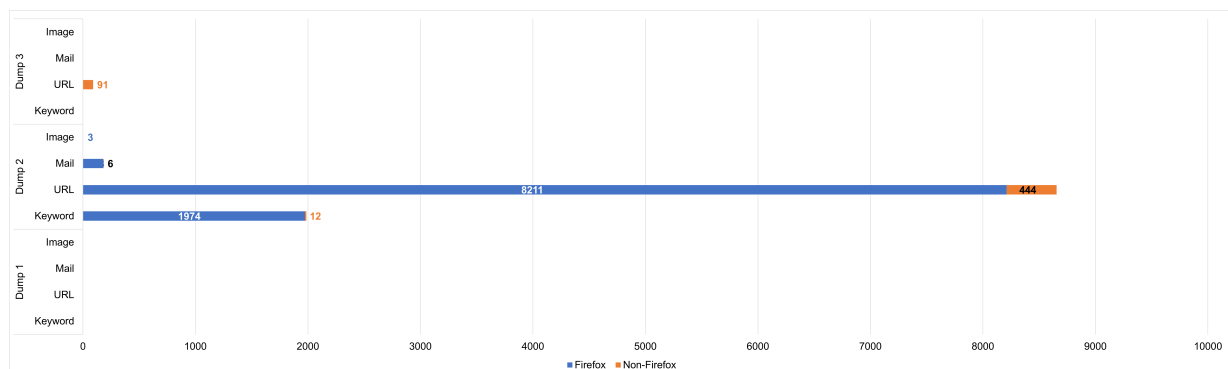


Abbildung 5.19: Summary

## Registry

Die Analyse der Registry zählt gemäß Methodik in Kapitel X sowohl zu den Common als auch Uncommon Locations

## Process Monitor SetValue Operations

Als Teil der Common Locations werden für Firefox alle Registry SSetValueSchreiboperationen der beiden Process Monitor Logfiles untersucht.

In beiden Logfiles wurden zwei Kategorien von Registry Keys geschrieben: "PreXULSkeletonUISettings" und "Business Activity Monitoring". In Abbildung X ist der Anteil der Schreiboperationen je Kategorie für beide Logfiles gezeigt.

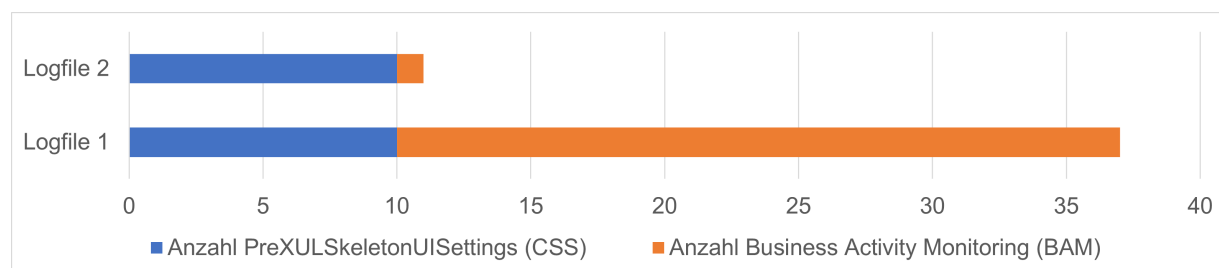


Abbildung 5.20: Comparison of found PB artifacts between RAM Dumps

**PreXULSkeletonUISettings** Der "PreXULSkeletonUISettings" Registry Key enthält Einstellungen für die Benutzeroberfläche (UI) des Firefox-Browsers, insbesondere für das sogenannte SSkeleton UI, eine vereinfachte Benutzeroberfläche, die während des Ladens des Browsers angezeigt wird, bevor die vollständige Benutzeroberfläche geladen ist. PreXULSkeletonUISettings Registry Keys haben das Format HKCU\SOFTWARE\Mozilla\Firefox\PreXULSkeletonUISettings\<Absoluter Firefox Installationspfad>\firefox.exe|<Skeleton UI Setting>. Somit enthält der Key den absoluten Installationspfad von Firefox gefolgt von einer Skeleton UI Einstellung. Nachfolgend sind alle möglichen UI Einstellungen aufgelistet, gefolgt vom Datentyp des Keys.

- ScreenX (DWORD)
- ScreenY (DWORD)
- Width (DWORD)
- Height (DWORD)
- Maximized (DWORD)
- Flags (DWORD)
- CssToDevPixelScaling (REG\_BINARY)
- UrlbarCSSSpan (REG\_BINARY)
- SearchbarCSSSpan (REG\_BINARY)
- SpringsCSSSpan (REG\_BINARY)

Somit enthalten die Keys nur Daten zur Formatierung und Struktur der grafischen Oberfläche. Es wurden keine PB Artefakte geschrieben

**Business Activity Monitoring** "Business Activity Monitoring", kurz BAM ist eine weitgehend undokumentierte Windows Funktion, die im Hintergrund ausgeführte Programme steuert. Der Registry Key hat das Format HKLM\System\CurrentControlSet\Services\bam\State\UserSettings\<SID>\Device\Harddisk\Firefox Installationspfad>\firefox.exe und den Datentyp REG\_BINARY. Jeder Schlüssel wird durch die Sicherheits-ID (SID) des Benutzers identifiziert. Ein BAM Registry Key schreibt für alle ausgeführten Programme — hier Firefox — den Zeitstempel der letzten Ausführung. PB Artefakte sind dabei nicht enthalten.

### Stringsuche in Registry Hives

Gemäß Methodik in Kapitel X wird die Firefox Registry als Uncommon Location behandelt, indem über alle auf der Festplatte vorhandenen Registry Datenbanken, den Registry-Hives, eine Stringsuche durchgeführt wird, ohne die Struktur der Hives zu beachten. Dazu wurden sowohl die System-Hives als auch die User-Hives aus Tabelle X (TODO!) aus jedem Snapshot extrahiert und mithilfe des Registry Explorers nach PB Artefakten durchsucht. Dabei wurde in keinem Snapshot in keinem Hive ein PB Artefakt gefunden.

\*\*\* TODO: Zusammenfassung Firefox \*\*\*



## 5.2 Tor

In diesem Abschnitt werden die Ergebnisse der Datenanalyse der Common Locations, Uncommon Locations sowie der Registry für den Tor-Browser präsentiert.

### Common Locations

Als Erstes werden die Common Locations analysiert, um potenzielle Hinweise auf Internetaktivitäten des Browsing Szenarios zu finden. Bei der Untersuchung der gängigen Speicherorte wird gemäß der im Kapitel X (TODO!) beschriebenen Methodik zwischen Schreibvorgängen in den Protokolldateien des Process Monitors und den SQLite-Datenbanken zur Verwaltung von Benutzerdaten unterschieden.

### Process Monitor WriteFile Operations

Bei der Versuchsdurchführung für den Tor-Browser gemäß Kapitel X wurden drei Process Monitor Logfiles erstellt. Diese Dateien enthalten alle aufgezeichneten Prozessaktivitäten während des Browsing Szenarios, nach dem Erzeugen einer "Neuen Identität" sowie nach Schließen des Browsers. Eine gemäß Methodik in Kapitel X verarbeitete Tabelle mit allen in den Logfiles identifizierten Dateien ist im Anhang X aufgeführt. Für jede Datei wurde vermerkt ob und wie sie wiederherstellbar war, mit welchem Tool die Datei analysiert wurde und ob PB Artefakte enthalten sind.

In Tabelle X (TODO!) sind die ausschließlich wiederherstellbaren Dateien aufgeführt. Die Dateien wurden in die vier Kategorien "Cache", "datareporting", und "Sonstige Dateien" eingeordnet. In keiner der identifizierten Dateien konnten PB Artefakte gefunden werden.

Kategorie	Datei	Logfile 1	Logfile 2	Logfile 3
Cache	I:\Cache\profile.default\startup\cache\startup\cache.8.little	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
Datareporting	I:\datareporting\gleandb\data.safe.bin	Keine PB Artefakte	Keine Schreiboperationen	Keine PB Artefakte
	I:\datareporting\state.json	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
SQLite	I:\storage\permanent\chrome\idb\3670112724\segment-es.sqlite	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I:\storage\permanent\chrome\idb\1657114535\Amc\tevt\sty.sqlite	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\places.sqlite	Keine PB Artefakte	Keine PB Artefakte	Keine PB Artefakte
	I\cookies.sqlite	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\storage.sqlite	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\avicons.sqlite	Keine PB Artefakte	Keine Schreiboperationen	Keine PB Artefakte
	I\webappsstore.sqlite	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\omhistory.sqlite	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\addon\startup.json.ls4	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\AlternateServices.txt	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
Sonstige Dateien	I\broadcast-listeners.json	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\extensions.json	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\extensionstaged\73a6fe31-595d-460b-a320-fcc0f8043232.xpi	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\omion-allases.json	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\prefs-1.jp	Keine PB Artefakte	Keine PB Artefakte	Keine PB Artefakte
	I\security_state\data.safe.bin	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\setting\data.safe.bin	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\SiteSecurity\ServiceState.txt	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\profile.default\session\checkpoints.json.tmp	Keine PB Artefakte	Keine Schreiboperationen	Keine Schreiboperationen
	I\profile.default\store.json	Keine PB Artefakte	Keine Schreiboperationen	Keine PB Artefakte
	I\profile.default\cert_override.txt	Keine Schreiboperationen	Keine PB Artefakte	Keine Schreiboperationen
	I\profile.default\enumarate_devices.txt	Keine Schreiboperationen	Keine PB Artefakte	Keine Schreiboperationen
	I\profile.default\session\checkpoints.json.tmp	Keine Schreiboperationen	Keine Schreiboperationen	Keine PB Artefakte
	I\storage\default\moz-extension+++3041a34e-916a-4fca-8ea0-53f966d7a1f1\userContextId=4294967295\metadata-v2	Keine Schreiboperationen	Keine Schreiboperationen	Keine PB Artefakte
	Caches			
	Profile Default			

Abbildung 5.21: Tabelle mit wiederherstellbaren Dateien: Logfile 1 vs. Logfile 2

Ähnlich wie bei der Analyse der Schreiboperationen von Firefox in Kapitel X (TODO!), konnten für den Tor-Browser zwei Pfade identifiziert werden, wo sich Dateien befinden, in die geschrieben wurde.

**Caches** C:\Users\Forensik\Desktop\Tor Browser\Browser\TorBrowser\Data\Browser\Caches\profile.def

**Profile.default** C:\Users\Forensik\Desktop\Tor Browser\Browser\TorBrowser\Data\Browser\profile.default\

In Tabelle X (TODO!) sind die Dateien je nach Speicherort "Caches"(Hellblau) oder "Profile.default"(Dunkelblau) eingefärbt.

Bei der Auswertung der Process Monitor Logfiles wurde festgestellt, dass alle Schreiboperationen von "firefox.exe" Prozessen durchgeführt wurde, nicht "tor.exe"

Obwohl keine der Dateien PB Artefakte enthält, werden zum vollständigen Browserverständnis im Sinne der White-Box-Forensik die wichtigsten Dateien im Zusammenhang des Tor-Browsers genauer untersucht.

**Cache** Der Tor-Browser schreibt eine einzige Cache-Datei \Caches\profile.default\startupCache\startupCache.8 little eine interne Datei, welche erstellt wird, um den Startvorgang des Browsers zu beschleunigen. Sie enthält Informationen über bereits geladene Browser-Komponenten wie JavaScript-Code, CSS-Dateien, Bilder und andere Ressourcen. Bei Untersuchung mit HxD konnten keine PB Artefakte gefunden werden.

**Datareporting** Im "DatareportingOrdner wird vom Tor-Browser die Datei \datareporting\data.safe.bin geschrieben. Sie enthält verschlüsselte und anonyme "Glean"-Informationen über die Nutzung des Browsers. In HxD konnten keine PB Artefakte gefunden werden. Weiterhin wird die Datei \datareporting\state.json geschrieben. Sie enthält Informationen über den Zustand und die Konfiguration des Tor-Browsers, beispielsweise installierte Add-Ons, oder Browser-Einstellungen. Sie wird verwendet, um dem Browser bei Bedarf den Zustand und die Einstellungen wiederherzustellen. Eine Analyse mit Notepad++ und dem JSON-Plugin brachte keine PB-Artefakte.

**Sonstige Dateien** Die im ersten Logfile geschriebene Datei AlternateServices.txt enthält onion URLs des HTTP Alternative Services ist ein Mechanismus. Dieser ermöglicht es Servern, Clients mitzuteilen, dass der Dienst, auf den sie zugreifen, an einem anderen Netzwerkstandort oder über ein anderes Protokoll verfügbar ist. Die Datei speichert diese Zuordnung.

Weiterhin wird während des Browsing Szenarios die Datei \extensions\staged\73a6fe31-595d-460b-a920-fcc0f8 geschrieben. Dabei handelt es sich um das von Tor verwendete "NoScriptAddOn zur selektiven Ausführung von JavaScript Webseiteninhalten. Nach Extraktion dieser Datei, kann diese per Drag-and-Drop in ein geöffnetes Firefox-Fenster gezogen werden und es ist möglich, die Erweiterung zu installieren.

Die geschriebene Datei onion-aliases.json enthält SSecureDrop-Adressen, beispielsweise für die Süddeutsche Zeitung. SecureDrop ist ein Open-Source-Software-Tool, das von Journalisten und Nachrichtenorganisationen verwendet wird, um anonyme Informationen von Whistleblowern entgegenzunehmen. Es ermöglicht den sicheren Austausch von Informationen, ohne die Identität der Quelle preiszugeben. Whistleblower können über .onion-URLs auf die SecureDrop-Websites zugreifen und vertrauliche Dokumente oder Nachrichten sicher und anonym übermitteln.

Schließlich wurde in die Datei SiteSecurityServiceState.txt geschrieben. Diese Datei speichert Daten wie Zertifikate, Verschlüsselungseinstellungen und andere Sicherheitsmerkmale, die von den besuchten Websites verwendet werden. Es ist anzumerken, dass diese Datei früher private Browsing

Artefakte enthält. In der aktuellen Tor-Browser-Version konnten keine private Browsing Artefakte gefunden werden.

## SQLite Datenbanken

Wie in Kapitel X (Methodik, TODO!) erwähnt, werden SQLite Datenbanken als Datenstrukturen für Nutzerdaten genauer untersucht. Mithilfe der Process Monitor Logfiles wurden die in Tabelle X dargestellten SQLite-Datenbanken für Firefox identifiziert:

Aus Process Monitor Logfiles ist erkennbar, dass Tor die gleichen SQLite Datenbanken wie Firefox aus Kapitel X (TODO!) verwaltet und beschreibt.

Wie bei der Analyse der SQLite-Datenbanken bei Firefox wird die Entwicklung von Dateinhalt in allen fünf Festplatten-Images der Snapshots 1, 2, 3-1, 3-2 und 4 betrachtet. Die Ergebnisse sind in Tabelle X (TODO!) dargestellt. Nach Browser-Installation wurde noch keine SQLite-Datei angelegt (Snapshot

File	Snapshot 1: Browser installation	Snapshot 2: After Browsing Scenario, Browser open		(For only) Snapshot 3-1: After Identity reset		Snapshot 3: After Browsing Scenario, Browser closed		Snapshot 4: VM Shutdown	
		Vor WAL	Nach WAL	Tor (Diff)	Nach WAL	Tor	Nach WAL	Vor WAL	Nach WAL
places.sqlite	N/A	Initialisiert, Daten: Onion URLs für Tor Standardseiten, wie "The Tor Blog" oder "Tor Browser Manual" und Spenden-Seite ( <a href="http://travelpod4b3idaz3yqzpm3h44bubm2dcsqzazg35de-united-europe">http://travelpod4b3idaz3yqzpm3h44bubm2dcsqzazg35de-united-europe</a> )	no diff	Indizes bei vorhandenen Seiten aktualisiert	no diff	Indizes bei vorhandenen Seiten aktualisiert	no diff	no diff	no diff
cookies.sqlite	N/A	Leer initialisiert (Nur Spaltennamen)	leer	leer	leer	leer	leer	leer	leer
storage.sqlite	N/A	Initialisiert, Einträge wie "Private Information", "Standard onion URL für Tor Standardseiten, wie "The Tor Blog" oder "Tor Browser Manual" und Spenden-Seite ( <a href="http://travelpod4b3idaz3yqzpm3h44bubm2dcsqzazg35de-united-europe">http://travelpod4b3idaz3yqzpm3h44bubm2dcsqzazg35de-united-europe</a> )	leer	leer	leer	leer	leer	leer	leer
favicons.sqlite	N/A	Initialisiert, Einträge wie "Private Information", "Standard onion URL für Tor Standardseiten, wie "The Tor Blog" oder "Tor Browser Manual" und Spenden-Seite ( <a href="http://travelpod4b3idaz3yqzpm3h44bubm2dcsqzazg35de-united-europe">http://travelpod4b3idaz3yqzpm3h44bubm2dcsqzazg35de-united-europe</a> )	no diff	no diff	no diff	in allen drei tabellen Indizes aktualisiert	no diff	no diff	no diff
webappstore.sqlite	N/A	Leer initialisiert (Nur Spaltennamen)	leer	leer	leer	leer	leer	leer	leer
formhistory.sqlite	N/A	Leer initialisiert (Nur Spaltennamen)	leer	leer	leer	leer	leer	leer	leer
ROSTH45584mcu4m38ty.sqlite	N/A	Leer initialisiert (Nur Spaltennamen)	leer	leer	leer	leer	leer	leer	leer
3870112124zozgmooitot-es.sqlite	N/A	Initialisiert, 1 Zeile: "origin: chrome"	no diff	no diff	no diff	gleich bleibt	no diff	no diff	no diff
		Leer							
		Unverändert							
		Neue (Fake-Icon) Inhalt							

Abbildung 5.22: Comparison of found PB artifacts between RAM Dumps

1).

Während des Browsing Szenarios wurden alle Datenbanken initialisiert. In places.sqlite wurden automatisch .onion URLs geschrieben, die zu Tor Standardseiten führen. Beispielsweise Seiten wie "The Tor Blog" oder "Tor Browser Manual". Die gleichen Einträge wurden in der favicons.sqlite Datenbank geschrieben, mit dem Präfix "Fake-favicon-uri". Ein tatsächliches Icon wie bei Firefox in Kapitel X wurde nicht in die Datenbank geschrieben. Weiterhin erhielt die "remote settings" Datenbank den gleichen Eintrag wie es bereits bei Firefox der Fall war. Der Eintrag enthält keine PB Artefakte. Die restliche SQLite-Dateien wurden ohne Inhalt angelegt, nur die Spaltennamen wurden definiert. Nach Durchführung der WAL Checkpoints bleiben Dateien unverändert.

Nachdem dem Tor-Browser eine "Neue Identität" zugewiesen wurde (Snapshot 3-1), wurden in places.sqlite die Indizes bei den eingetragenen Seiten aktualisiert. Die restlichen Dateien blieben unverändert. Das Schreiben der WAL-Dateien in die Hauptdatenbanken veränderte den Inhalt nicht.

Nach Schließen des Browsers (Snapshot 3) wurden in places.sqlite sowie favicons.sqlite erneut Indizes bei eingetragenen Seiten aktualisiert. Die restliche Dateien blieben unverändert, ebenso ergaben die WAL Checkpoints keine Veränderungen.

Nach Herunterfahren der VM (Snapshot 4) blieben alle Datenbanken unverändert. Auch nach Durchführung der WAL Checkpoints gab es keine neuen Inhalte.

Im Balkendiagramm X (TODO!) ist zu erkennen, dass die meisten Schreiboperationen im ersten Logfile stattfinden. Dort werden Dateien jeder Kategorie beschrieben. Das Schließen des Tor-Browsers führt zu mehr oder genauso vielen Schreiboperationen wie das Zuweisen einer "Neuen Identität". Keine der geschriebenen Dateien enthielt private Browsing Artefakte. \*\*\* TODO: Was noch? \*\*\*

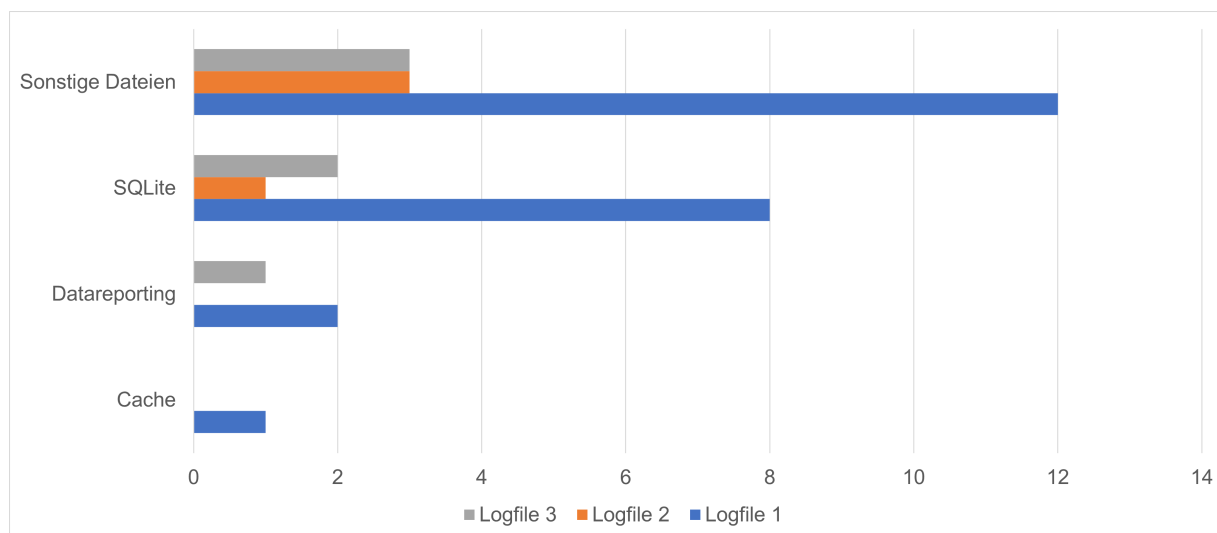


Abbildung 5.23: Comparison of found PB artifacts between RAM Dumps

## Uncommon Locations

Nachfolgend werden die Analyseergebnisse der Firefox Uncommon Locations beschrieben. Wie in Kapitel X erläutert, wird im Gegensatz zu Common Locations die Suchrichtung umgekehrt und es werden alle gesammelten Daten nach einem spezifischen PB Artefakt durchsucht. Somit benötigt ein Forensiker kein Wissen über das Browserverhalten. Stattdessen wird sich auf die Vollständigkeit der Funktionen von Forensik-Tools verlassen. Im Rahmen dieses Versuchs werden die Tools Autopsy und Volatility verwendet.

## Analyse mit Autopsy

Bei den Common Locations in Kapitel X wird Autopsy nur zur Dateixtraktion genutzt. Im Falle der Uncommon Locations dient Autopsy als forensisches Werkzeug zur Datenanalyse.

Bei White-Box Analyse/Common Locations: Autopsy nur zur Dateixtraktion genutzt, hier: als konkretes forensisches Werkzeug

Stichwortsuche: - In allen Snapshots keine Treffer (auch innerhalb \$Carved) - TODO: Pagefile gefunden?

Von Autopsy automatisch indexierte Dateien: In allen Fällen: keine Dateien gelöscht, nur über Zeitraum der Snapshots neue dazugekommen - Web Bookmarks: Snapshot 1: > Bing.url (Unter C:/User/Forensik/Favorites/Links) enthält Bing Startseite Snapshot 2: > unverändert zu 1 Snapshot


Source Name	S	C	O	URL	Title	▼ Date Created	Program Name	Domain
 Bing.url			19	http://go.microsoft.com/fwlink/p/?LinkId=255142	Bing.url	2023-04-25 16:09:28 MESZ	Internet Explorer Analyzer	microsoft.com

Abbildung 5.24: Autopsy Web Bookmarks

3-1: > unverändert zu 2 Snapshot 3-2: > unverändert zu 3-1 Snapshot 4: > unverändert zu 3-2 - Web Cookies: Snapshot 1: > 9 Einträge in WebCacheV01.dat (= DB des Internet Explorers zum speichern










Source Name	S	C	O	URL	▼ Date Accessed	Name	Value	Program Name	Domain
 WebCacheV01.dat			15	bing.com	2023-05-07 13:56:11 MESZ	SUID	A	Microsoft Edge Analyzer	bing.com
 WebCacheV01.dat			15	bing.com	2023-05-07 13:47:01 MESZ	ANON	A=D8530A977A1F5DAD20B78D8CFFFFFFF	Microsoft Edge Analyzer	bing.com
 WebCacheV01.dat			15	login.live.com	2023-05-07 12:26:40 MESZ	__Host-M5AAUTHP		Microsoft Edge Analyzer	live.com
 WebCacheV01.dat			15	live.com	2023-05-07 12:25:56 MESZ	MUID	0EDC58E500A76FCE1B0F4BEF04A76BD6	Microsoft Edge Analyzer	live.com
 WebCacheV01.dat			15	www.bing.com	2023-05-07 12:25:44 MESZ	MUIDB	31708C5FC3CF47068AFADC1CB47D0111	Microsoft Edge Analyzer	bing.com
 WebCacheV01.dat			15	bing.com	2023-05-07 12:25:44 MESZ	SRCHD	AF=NIFORM	Microsoft Edge Analyzer	bing.com
 WebCacheV01.dat			15	bing.com	2023-05-07 12:25:44 MESZ	SRCHUID	V=28GUID=62F5FD78E9D944468AFDF9DEC81881038dm...	Microsoft Edge Analyzer	bing.com
 WebCacheV01.dat			15	bing.com	2023-05-07 12:25:44 MESZ	SRCHUSR	DOB=20230507	Microsoft Edge Analyzer	bing.com
 WebCacheV01.dat			15	bing.com	2023-05-07 12:25:44 MESZ	SRCHHPGUSR	SRCHLANG=de	Microsoft Edge Analyzer	bing.com

Abbildung 5.25: Autopsy Web Cookies

von Browserdaten): Cookies für bing.com und live.com (= outlook) Snapshot 2: > unverändert zu 1 Snapshot 3-1: > unverändert zu 2 Snapshot 3-2: > unverändert zu 3-1 Snapshot 4: > unverändert zu 3-2 - Web History: Snapshot 1: > 2 Einträge in WebCacheV01.dat: - 2x live.com (= outlook) Snapshot






Source Name	S	C	O	URL	▼ Date Accessed	Program Name	Domain	Username
 WebCacheV01.dat				file:///Z:/Logfile2-2	2023-05-07 14:28:06 MESZ	Microsoft Edge Analyzer		Forensik
 WebCacheV01.dat				file:///Z:/Logfile2-1	2023-05-07 14:17:05 MESZ	Microsoft Edge Analyzer		Forensik
 WebCacheV01.dat				file:///Z:/Logfile1	2023-05-07 13:55:54 MESZ	Microsoft Edge Analyzer		Forensik
 WebCacheV01.dat			15	https://login.live.com/oauth20_authorize.srf?client_id=000...	2023-05-07 12:26:43 MESZ	Microsoft Edge Analyzer	live.com	Forensik
 WebCacheV01.dat			15	https://login.live.com/oauth20_desktop.srf?lc=1031	2023-05-07 12:26:40 MESZ	Microsoft Edge Analyzer	live.com	Forensik

Abbildung 5.26: Autopsy Web History

2: > 1 neuer Einträge in WebCacheV01.dat: - file:///Z:/Logfile\_1 (= Process Monitor Logfile, die in shared-Folder geladen wurde) -> Erklärung? Snapshot 3-1: > 1 neuer Eintrag in WebCacheV01.dat: - file:///Z:/Logfile\_2-1 (= Process Monitor Logfile, die in shared-Folder geladen wurde) -> Erklärung? Snapshot 3-2: > 1 neuer Eintrag in WebCacheV01.dat: - file:///Z:/Logfile\_2-2 (= Process Monitor Logfile, die in shared-Folder geladen wurde) -> Erklärung? Snapshot 4: > unverändert zu 3-2 - Web Categories: Snapshot 1: > 2x WebCacheV01.dat aufgelistet => Mit HxD untersucht, keine



Source Name	▲ S	C	O	Source Type	Score	Conclusion	Configuration	Justification	Domain	Host	Name
 WebCacheV01.dat			0	File	Unknown				bing.com	bing.com	Search Engine
 WebCacheV01.dat			0	File	Unknown				live.com	login.live.com	Web Email

Abbildung 5.27: Autopsy Web Categories

PB Artefakte Snapshot 2: > unverändert zu 2 Snapshot 3-1: > unverändert zu 3 Snapshot 3-2: > unverändert zu 3-1 Snapshot 4: > unverändert zu 3-2

Zusammenfassung: - keine PB Artefakte - Keine neuen Erkenntnisse vgl. mit intensiver Analyse mittels Process Monitor in Kapitel X - .onion URL Einträge in places.sql nicht erkannt

## Analyse mit Volatility

Nachdem die Firefox Festplattenabbilder als Uncommon Location mit Autopsy untersucht wurden, werden nachfolgend die Analyseergebnisse des RAMs als Uncommon Location beschrieben. Dazu wurde eine Stringsuche im gesamten RAM nach PB Artefakten durchgeführt. Wie in Kapitel X ausführlich beschrieben muss ein gefundener String eindeutig einem Browser zugeordnet werden können.

Deshalb wurde dazu das Volatility PlugIn "Yarascan" verwendet, ein Werkzeug um nach bestimmten Mustern im RAM zu suchen. Dazu wurden die in Tabelle X aufgeführten Yara-Regeln verwendet. Wie in Kapitel Methodik (TODO!) beschrieben, wird davon ausgehend das PlugIn "pslist" verwendet, um den Prozessnamen anhand PID zu identifizieren. Die Ergebnisse dieser Stringsuche sind nachfolgend nach Kategorie geordnet.

Vorgehen: Siehe "Methodik" Kapitel - Ausgangslage: Volatility Yarascan Treffer - Für jeden Treffer: virtueller Offset des Strings, PID, getriggerte Yararule, getriggerte Yara Component z(= Variablenname des gesuchten Strings), gefundener String - Neue Spalte: "Prozessname" zu jeder PID Prozessnamen - Ergebnisse Aufbereitet nach folgendem Schema: > Für jeden RAM Dump > Für jede Yararule > Für jede Component > Filter: Prozessname = Firefox -> Anzahl zählen > Filter: Prozessname = Alle Prozesse außer Firefox -> Anzahl zählen

Wie bei Firefox: HTML Artefakte wurden in keinem RAM Dump gefunden => Nicht aufgeführt

Yararule "Keyword": Analyse: > Ausschließlich in RAM Dump 2 und RAM Dump 3-1 Keyword Artefakte

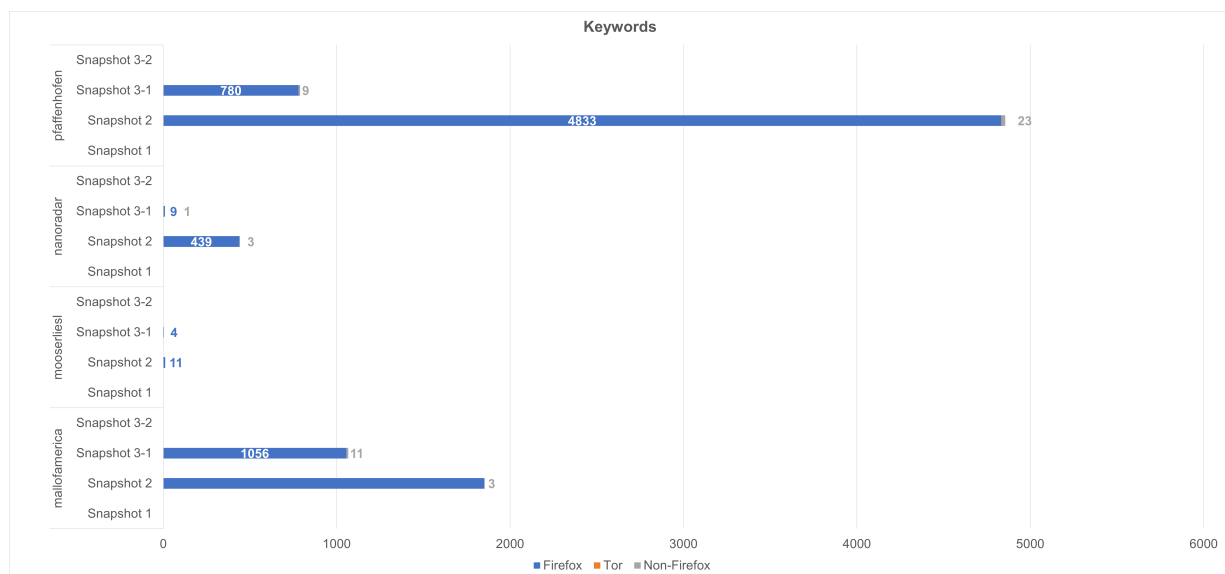


Abbildung 5.28: Keywords

gefunden > In RAM Dump 3-1 bei jedem Keyword deutlich weniger Artefakte als in RAM Dump 2 => Identitäts-Reset reduziert Keyword Artefakte deutlich > Hauptsächlich in Firefox Prozess, kein Artefakt in Tor.exe Prozess > Mit 4833 Artefakten in RAM Dump 2 am häufigsten "pfaffenhofen" vertreten. Vermutung: Evtl. weil Google Maps viele zusätzliche Artefakte lädt. > Nach Schließen von Tor Browser: keine Keyword Artefakte mehr in RAM

Yararule ÜRL": Analyse: > Wie bei Yararule "Keyword": Ausschließlich in RAM Dump 2 und RAM

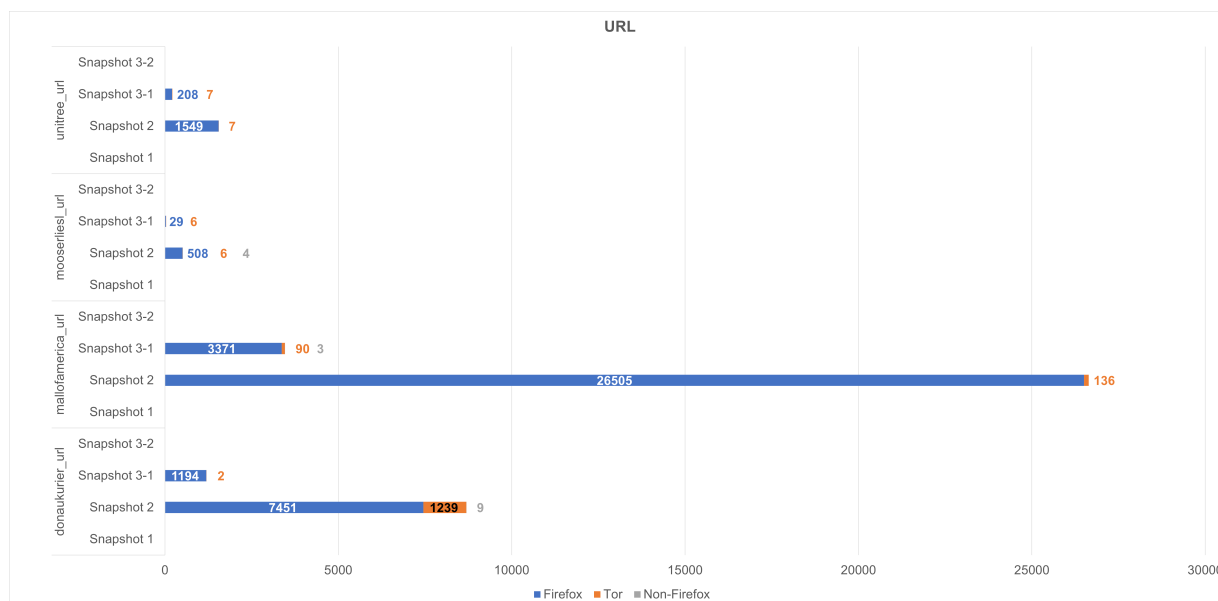


Abbildung 5.29: URL

Dump 3-1 Keyword Artefakte gefunden > In RAM Dump 3-1 bei jedem Keyword deutlich weniger Artefakte als in RAM Dump 2 => Identitäts-Reset reduziert URL Artefakte deutlich > Hauptsächlich in Firefox Prozess, danach am häufigsten Tor.exe Prozess und am wenigsten Artefakte in anderen Prozessen > Bemerkenswert: "mallofamerica.com" ist mit 26.505 mal in RAM Dump 2 am häufigsten als Artefakt gefunden worden. Vergleich: "mooserlies.de" wurde nur 508 mal in RAM Dump 2 gefunden > Nach Schließen von Tor Browser: keine URL Artefakte mehr in RAM

> TODO: DNSCache?

Yararule "Mail": Analyse: > Alle Mail Artefakte gefunden > Artefakte ausschließlich in Firefox Prozess gefunden > Artefakte fast ausschließlich in RAM Dump 2 Mail gefunden > Nur die Absenderadresse "computerforensikvl@gmail.com" wurde nach Identitäts-Reset in RAM Dump 3-1 gefunden > Absenderadresse ist häufigstes Mail Artefakt > Bemerkenswert: Passwort wurde 2x als Klartext im RAM gefunden! String Kontext: Offsets: PIDs: 0xb9ce29180c8 7420 0x2859f4ffd4e0 7420 0x24083b41858 8424 0x240840e5b08 8424

Yararule "Image": Analyse: > Hex-Wert von Donaukurier Bild wurde ein einziges mal im 2. RAM Dump in einem Firefox Prozess gefunden

Zusammenfassung = Stacked Bar Chart: - PB Artefakte ausschließlich in RAM Dump 2 und 3-1 gefunden - Nach Identitäts-Reset deutlich weniger Artefakte vorhanden - Am meisten URL-Artefakte gefunden, wobei mallofamerica.com dominant - HTML Artefakte wurden in keinem RAM Dump gefunden

TODO: Kreisdiagramme/Balkendiagramme mit Gesamtzahl an (Non-)Firefox Yarascan-Treffer erst im Vergleich mit Tor

Literatur:

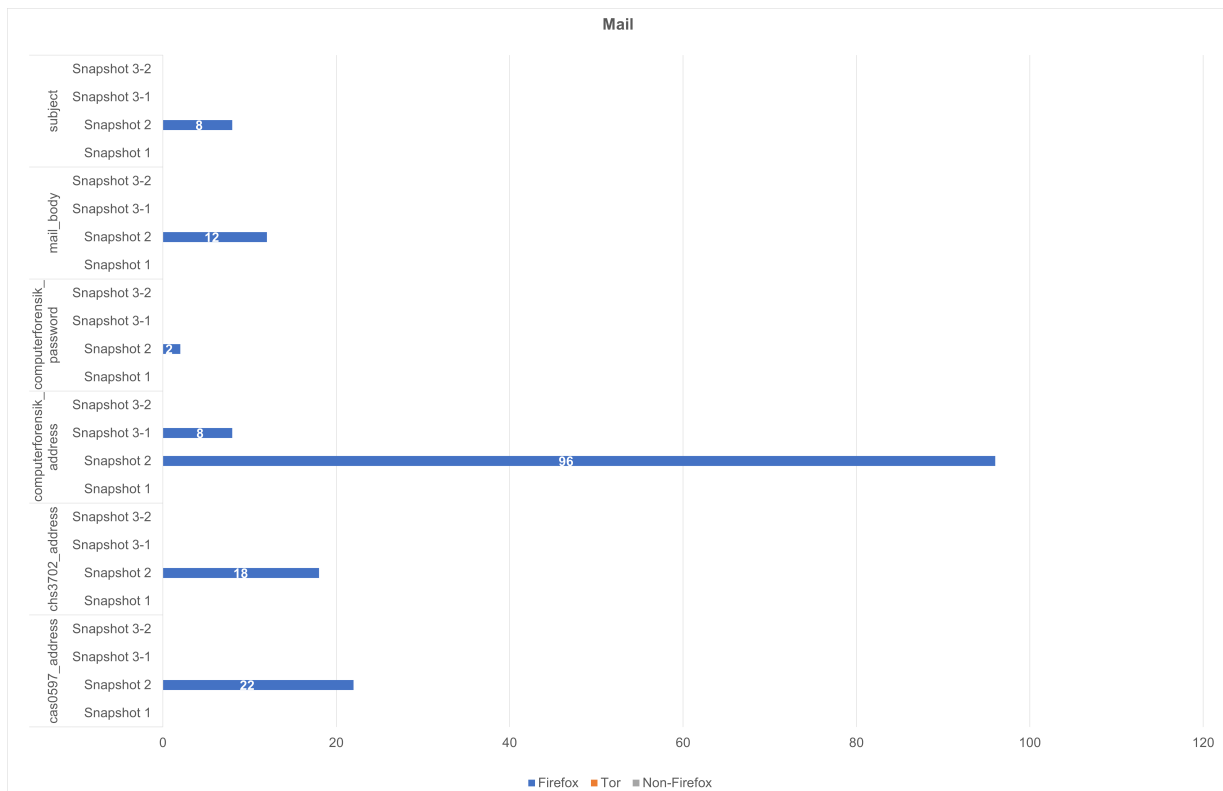


Abbildung 5.30: Mail

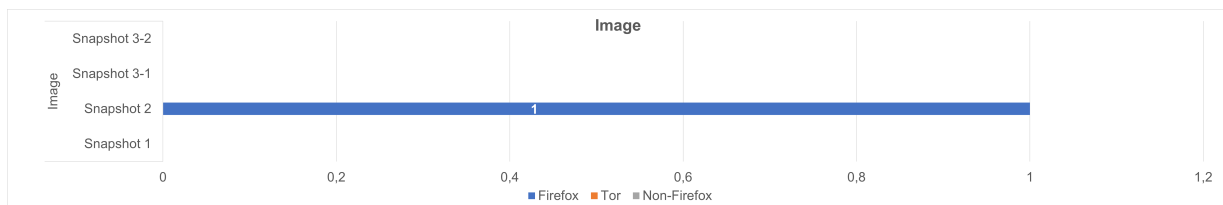


Abbildung 5.31: Image

## Registry

Die Analyse der Registry zählt gemäß Methodik in Kapitel X sowohl zu den Common als auch Uncommon Locations

### Process Monitor SetValue Operations

Als Teil der Common Locations werden für Firefox alle Registry SSetValueSSchreiboperationen der beiden Process Monitor Logfiles untersucht.

> Process Monitor: SetValue Operationen von Browser Kategorien Registry Keys: Analog zu Firefox  
 1) PreXULSkeletonUISettings: > Prefix: Absoluter Installationspfad von Firefox > Skeleton UI Einstel-



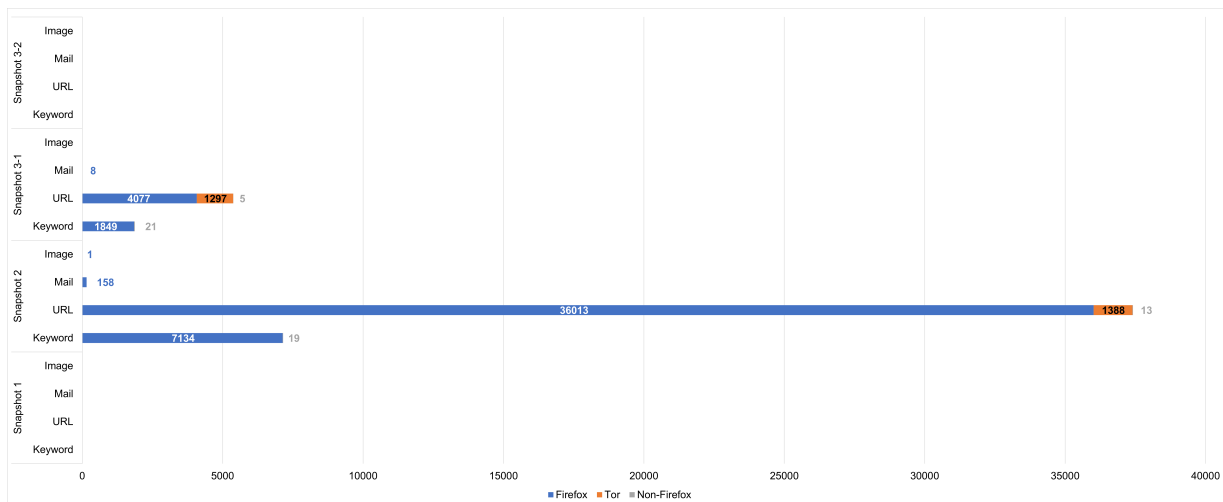


Abbildung 5.32: Summary

lungen von Firefox Definition: > Der "PreXULSkeletonUISettings"Registry Key enthielt Einstellungen für die Benutzeroberfläche (UI) des Firefox-Browsers, insbesondere für das sogenannte SSkeleton UI". Das Skeleton UI ist eine vereinfachte Benutzeroberfläche, die während des Ladens des Browsers angezeigt wird, bevor die vollständige Benutzeroberfläche geladen ist. Es besteht aus grundlegenden Steuerelementen und Elementen, die dem Benutzer die Interaktion ermöglichen, während der Rest der Benutzeroberfläche noch geladen wird. > Der "PreXULSkeletonUISettingsSchlüssel enthielt Konfigurationsoptionen wie Farben, Positionen und andere Einstellungen für das Skeleton UI. Durch das Bearbeiten dieses Schlüssels konnten Benutzer die Darstellung des Skeleton UI anpassen. Es ist jedoch wichtig zu beachten, dass das Ändern der Registrierungseinträge ein fortgeschrittenes Verfahren ist und Fehler zu Problemen mit dem Browser führen kann.

> Struktur der Keys: > Unterschiedliche UI Einstellungen - - - - - > keine PB Artefakte unter UI Einstellungen 2) Business Activity Monitoring > Quelle: > BAM is a mostly undocumented feature that controls the programs executed in the background. DAM is a feature for devices supporting the "Connected Standby" mode (i.e when a device is turned on, but its display will be turned off). As a result, the BAM registry keys will contain data on any devices, while DAM registry keys will only contain data on mobile devices. > The BAM registry key contains multiple subkeys under bam State

UserSettings, with one subkey per user, identified with the user SID. While the key is in the SYSTEM registry hive, program executions can thus still be tied to a specific user using this SID. > Each user-specific key contains a list of executed programs, with their full path and timestamp of last execution. > If a file is deleted, the eventual associated entry in the BAM is deleted as well after the system reboot. Additionally, BAM entries older than 7 days are deleted upon system boot. The BAM thus provides limited information on historic execution of programs > No entries are created in the BAM keys for executables on removable media and/or on network shares. > Key:

Quantitativ: (Diagramme) - Stacked Balkendiagramm jeweils für Logfile 1 und Logfile2: Anteil Kategorie 1 bzw.2 an allen Registry-Schreiboperationen

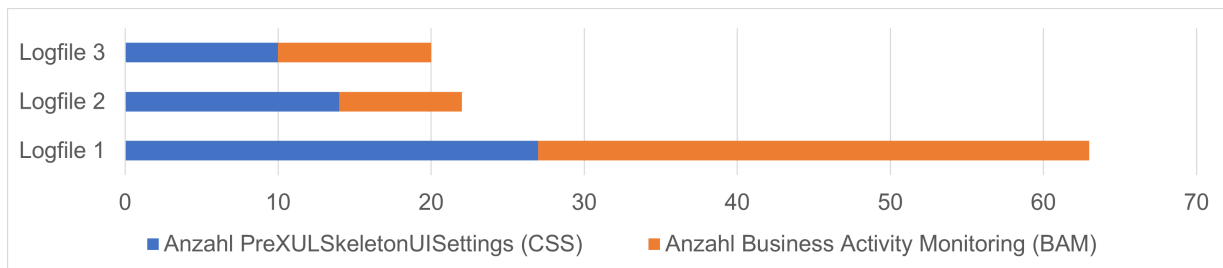


Abbildung 5.33: Comparison of found PB artifacts between RAM Dumps

### Stringsuche in Registry Hives

Gemäß Methodik in Kapitel X wird die Firefox Registry als Uncommon Location behandelt, indem über alle auf der Festplatte vorhandenen Registry Datenbanken, den Registry-Hives, eine Stringsuche durchgeführt wird, ohne die Struktur der Hives zu beachten. Dazu wurden sowohl die System-Hives als auch die User-Hives aus Tabelle X (TODO!) aus jedem Snapshot extrahiert und mithilfe des Registry Explorers nach PB Artefakten durchsucht. Dabei wurde in keinem Snapshot in keinem Hive ein PB Artefakt gefunden.

> Stringsuche in Registry Hives mit Registry Explorer (Siehe Liste) In allen Hives kein Treffer für alle Suchbegriffe

## 5.3 Chrome

### Uncommon Locations

o Autopsy Keyword-Suche: > Chrome and Edge produced five artefacts as reported by both tools. (FTK, Autopsy) [5] –> Artefakte werden nicht genannt! > only two temporary files (Figure 7) were recovered with Minitool Power Data Recovery but it was a dead end; Location: appdata/.../Chrome/.../Preferences/RF1533fa.TMP [4] > pagefile.sys file showed no traces at all [22]

## 5.4 Brave

## 6 Vergleich der Browser

- Zusammenfassung: Vergleich Tor v. Firefox und Brave v. Chrome

Firefox vs Tor: > Gestacktes Balkendiagramm zu veränderten SQLite DBs => Erst bei Vergleich mit Tor!

SSessionstore-Backup"fehlt in Tor

TODO: Kreisdiagramme/Balkendiagramme mit Gesamtzahl an (Non-)Firefox Yarascan-Treffer erst im Vergleich mit Tor

- Firefox v. Chrome (SStandardbrowser") - Tor v. Brave (SSichere Browser") - Zum Schluss: Eine große Tabelle"mit den wichtigsten Kategorien?

## 7 Diskussion

> Artefakte im DNS Cache: [24] ■ DNS-Caching ist eine Bedrohung für private Browsing ■ Diese Schwachstelle entsteht, weil das Betriebssystem DNS-Anfragen des Browsers im Cache speichert, unabhängig davon, ob der Browser im privaten Modus ist oder nicht ■ Mehrere Jahre nach der Meldung dieser Schwachstelle besteht sie immer noch in allen Browsern fort ■ Es wurden einige Erweiterungen von Drittanbietern entwickelt, um dieses Problem zu beheben, aber keine davon wurde von den Browserherstellern übernommen.

> Viele RAM-Artefakte - Firefox [15] ■ Darcie et al. (2014) fanden Beweise für das Web-Browsing in Form von JPEG- und HTML-Dateien in Live-Forensik, aber eine statische Forensik war erfolglos. ■ Eine vorherige Live-Forensik-Analyse des Firefox-Browsers zeigte, dass Artefakte aus einer privaten Browsing-Sitzung aus dem Speicher wiederhergestellt werden konnten. (Findlay and Leimich, 2014).

> IE hinterlässt viele Spuren im Gegensatz zu Ergebnissen: [13] o hidden folders are usually stored at C:/Users/User/AppData o evidence searches are conducted extensively in the C: partition o bookmarks remain and can be viewed o downloads remain in the downloads folder until the user manually deletes them o CacheView trace entire URL and browsing histories including the temporary files CacheView enables to find the image's URL and from specific website

> Urteil über die Privatheit von Tor nach [15] The design aim of preventing Tor from writing to disk (Perry et al., 2018) is not achieved in this version. ■ Configuration files, downloaded files, and browserrelated data are recoverable from the file system. ■ Significant data-leakage from the browsing session occurred: HTTP header information, titles of web pages and an instance of a URL were found in registry files, system files, and unallocated space. ■ The data-leakage contained the German word for 'search' in reference to a Google search. This hints at the locale of the Tor server used to exit the network (exit relay). The Tor Project's design aim of enabling secure deletion of the browser (Sandvik, 2013) is not achieved in this version. ■ References to: the installation directory, Firefox SQLite files, bridging IPs/ports, default bookmarks, Tor-related DLLs and Tor product information were all recovered after the browser was deleted. ■ In a scenario where the operating system paged memory, an instance

Weiterführende Arbeiten: > Cross-mode interference [9]: o the Chrome://memory page displays all the opened tabs in the browser regardless if they are in the usual or private mode -> Nicht mehr aktuell -> Stattdessen: Chrome Task-manager (Ctrl + Esc), Funktioniert auch bei Firefox > Unser Scope: Process Monitor nach Prozessnamen gefiltert - Weiterführend: Nach Pathnamen filtern: "Common Locations"

> Für wen wird Browser entwickelt > Warum und für wen wird Private Browsing analysiert? > Ist das Auffinden privater Browsing Artefakte Schuld von Browser Entwicklern? (Oder Schuld des Betriebssystems, wie in (TODO!) erwähnt)

> bei Process Monitor nur nach Browser-Prozessen gefiltert

> Warum in Literatur nur Windows untersucht?

Tor: Unsere Mission: Menschenrechte und Freiheiten durch die Entwicklung und Verbreitung von Open Source Anonymitäts- und Privatsphäre-Technologien zu fördern, ihre ungehinderte Verfügbarkeit zu unterstützen und ihr Verständnis in Wissenschaft und der Allgemeinheit zu vergrößern."

## 8 Fazit

Einleitend werden Struktur, Motivation und die abgeleiteten Forschungsfragen diskutiert.

# **Appendices**

All File Operations Firefox

LOGFILE 1:				
	Dateistatus	Verwendetes Tool zur Analyse	Enthaltenes Artefakte	
Cache	Dane vorhanden	Modul:acheView	Keine PB Artefakte	
	Dane vorhanden	Modul:acheView	Keine PB Artefakte	
	Dane vorhanden	Modul:acheView	Keine PB Artefakte	
	Dane vorhanden	Modul:acheView	Keine PB Artefakte	
	Dane vorhanden	Modul:acheView	Keine PB Artefakte	
Datei:reporting	Dane vorhanden	Modul:acheView	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	HxD	Keine PB Artefakte	
	Dane nicht wiederherstellbar	N/A	N/A	
	Nicht-Hilfsdatei verwendet	sql33 Kommandozeile	Keine PB Artefakte	
	Dane vorhanden	SQL:as Viewer	Keine PB Artefakte	
SQLite	Dane vorhanden	sql33 Kommandozeile	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	SQL:as Viewer	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	sql33 Kommandozeile	Keine PB Artefakte	
	Dane vorhanden	SQL:as Viewer	Keine PB Artefakte	
	Dane vorhanden	SQL:as Viewer	Keine PB Artefakte	
Sessionstore	Nicht-Hilfsdatei verwendet	dejeont4 - Nonpad++	Keine PB Artefakte	
	Dane vorhanden	HxD	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	Nonpad++	Keine PB Artefakte	
	Dane nicht wiederherstellbar	N/A	N/A	
	Dane nicht wiederherstellbar	N/A	Keine PB Artefakte	
LOGFILE 2:				
	Dateistatus	Verwendetes Tool zur Analyse	Enthaltenes Artefakte	
Cache	Dane vorhanden	Modul:acheView	Keine PB Artefakte	
	Dane vorhanden	Modul:acheView	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	HxD	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	N/A	N/A	
	Nicht-Hilfsdatei verwendet	N/A	N/A	
Datei:reporting	Dane vorhanden	SQL:as Viewer	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	sql33 Kommandozeile	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	sql33 Kommandozeile	Keine PB Artefakte	
	Dane vorhanden	SQL:as Viewer	Keine PB Artefakte	
	Dane vorhanden	SQL:as Viewer	Keine PB Artefakte	
SQLite	Dane vorhanden	sql33 Kommandozeile	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	SQL:as Viewer	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	dejeont4 - Nonpad++	Keine PB Artefakte	
	Dane vorhanden	HxD	Keine PB Artefakte	
	Nicht-Hilfsdatei verwendet	Nonpad++	Keine PB Artefakte	
Sessionstore	Dane nicht wiederherstellbar	N/A	N/A	
	Dane nicht wiederherstellbar	N/A	N/A	
	Dane nicht wiederherstellbar	N/A	N/A	
	Dane nicht wiederherstellbar	N/A	N/A	
	Dane nicht wiederherstellbar	N/A	N/A	

Abbildung .1: All File Operations Firefox: Logfile 1 vs. Logfile 2



## All File Operations Tor

[illegible]

Abbildung .2: All File Operations Firefox: Logfile 1 vs. Logfile 2 vs. Logfile 3

# Literatur

- [1] Gaurav Aggarwal u. a. "An Analysis of Private Browsing Modes in Modern Browsers." In: *USENIX security symposium*. 2010, S. 79–94.
- [2] Gabriele Bonetti u. a. "Black-box forensic and antforensic characteristics of solid-state drives". In: *Journal of Computer Virology and Hacking Techniques* 10 (2014), S. 255–271.
- [3] Howard Chivers. "Private browsing: A window of forensic opportunity". In: *Digital Investigation* 11.1 (2014), S. 20–29.
- [4] Hasan Fayyad-Kazan u. a. "Forensic analysis of private browsing mechanisms: Tracing internet activities". In: (2021).
- [5] Ryan M Gabet, Kathryn C Seigfried-Spellar und Marcus K Rogers. "A comparative forensic analysis of privacy enhanced web browsers and private browsing modes of common web browsers". In: *International Journal of Electronic Security and Digital Forensics* 10.4 (2018), S. 356–371.
- [6] Ms Pooja Gupta. "Capturing Ephemeral Evidence Using Live Forensics". In: *IOSR J. Electron. Commun. Eng* (2013), S. 109–113.
- [7] Meenu Hariharan, Akash Thakar und Parvesh Sharma. "Forensic Analysis of Private Mode Browsing Artifacts in Portable Web Browsers Using Memory Forensics". In: *2022 International Conference on Computing, Communication, Security and Intelligent Systems (IC3SIS)*. IEEE. 2022, S. 1–5.
- [8] Nihad A Hassan. *Digital forensics basics: A practical guide using Windows OS*. Apress, 2019.
- [9] Ashley Hedberg. *The privacy of private browsing*. Techn. Ber. Technical Report, Tufts University, MA, USA, 2013.
- [10] Graeme Horsman u. a. "A forensic examination of web browser privacy-modes". In: *Forensic Science International: Reports* 1 (2019), S. 100036.
- [11] Aina Izzati und Nurul Hidayah Ab Rahman. "A Comparative Analysis of Residual Data Between Private Browsing and Normal Browsing Using Live Memory Acquisition". In: *Applied Information Technology And Computer Science* 3.2 (2022), S. 68–83.
- [12] Ahmed Redha Mahlous und Houssam Mahlous. "Private Browsing Forensic Analysis: A Case Study of Privacy Preservation in the Brave Browser". In: *International Journal of Intelligent Engineering Systems* 13.06 (2020), S. 294–306.
- [13] Raihana Md Saidi u. a. "Analysis of Private Browsing Activities". In: *Regional Conference on Science, Technology and Social Sciences (RCSTSS 2016) Theoretical and Applied Sciences*. Springer. 2018, S. 217–228.

- 
- [14] Reza Montasari und Pekka Peltola. "Computer forensic analysis of private browsing modes". In: *Global Security, Safety and Sustainability: Tomorrow's Challenges of Cyber Security: 10th International Conference, ICGS3 2015, London, UK, September 15-17, 2015. Proceedings 10*. Springer. 2015, S. 96–109.
- [15] Matt Muir, Petra Leimich und William J Buchanan. "A forensic audit of the tor browser bundle". In: *Digital Investigation* 29 (2019), S. 118–128.
- [16] Apurva Nalawade, Smita Bharne und Vanita Mane. "Forensic analysis and evidence collection for web browser activity". In: *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*. IEEE. 2016, S. 518–522.
- [17] Junghoon Oh, Seungbong Lee und Sangjin Lee. "Advanced evidence collection and analysis of web browser activity". In: *Digital investigation* 8 (2011), S62–S70.
- [18] Donny Jacob Ohana und Narasimha Shashidhar. "Do private and portable web browsers leave incriminating evidence? a forensic analysis of residual artifacts from private and portable web browsing sessions". In: *2013 IEEE Security and Privacy Workshops*. IEEE. 2013, S. 135–142.
- [19] Daniel Perdices u. a. "Web browsing privacy in the deep learning era: Beyond VPNs and encryption". In: *Computer Networks* 220 (2023), S. 109471.
- [20] Digvijaysinh Rathod. "Darknet forensics". In: *future* 11 (2017), S. 12.
- [21] Tri Rochmadi, Imam Riadi und Yudi Prayudi. "Live forensics for anti-forensics analysis on private portable web browser". In: *Int. J. Comput. Appl* 164.8 (2017), S. 31–37.
- [22] Huwida Said u. a. "Forensic analysis of private browsing artifacts". In: *2011 International Conference on Innovations in Information Technology*. IEEE. 2011, S. 197–202.
- [23] Priya P Sajan u. a. "Tor Browser Forensics". In: *Turkish Journal of Computer and Mathematics Education (TURCOMAT)* 12.11 (2021), S. 5599–5608.
- [24] Kiavash Satvat u. a. "On the privacy of private browsing—a forensic approach". In: *Data Privacy Management and Autonomous Spontaneous Security: 8th International Workshop, DPM 2013, and 6th International Workshop, SETOP 2013, Egham, UK, September 12-13, 2013, Revised Selected Papers*. Springer. 2014, S. 380–389.
- [25] Yunus Yusoff, Roslan Ismail und Zainuddin Hassan. "Common phases of computer forensics investigation models". In: *International Journal of Computer Science & Information Technology* 3.3 (2011), S. 17–31.