National College of Ireland

Dissecting Private Browsing in Android Devices

MSc Research Project
Academic Internship

Eduards Samuls
21244545

School of Computing
National College of Ireland

Supervisor: Evgeniia Jayasekera

National College of Ireland
MSc Project Submission Sheet
School of Computing

National College of Ireland

Student Name: Eduards Samuls.....

Student ID: 21244545.....

Programme: MSc CyberSecurity..... Year: 1.....

Module: Academic Internship.....

Supervisor: Evgeniia Jayasekera.....

Submission Due Date: 18 September 2023.....

Project Title: Dissecting Private Browsing in Android Devices.....

Word Count: 9125..... Page Count: 22.....

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Eduards Samuls.....

Date: 16 September 2023.....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Dissecting Private Browsing in Android Devices

Eduards Samuls
21244545

Abstract

This paper forensically investigates the private browsing modes of Brave, Chrome, Firefox and Tor browsers, examining the information retained on disk and RAM. The investigation was necessary as it was identified that no android research covered the area of browser private windows. HTTP connections sent to developers are also examined to identify if usage information or other identifiers are sent back. The investigation reveals that on android all tested browsers except Tor retain at least part of information on disk and/or RAM with and without addons. Further, it explains that Chrome and Firefox send identifiers back to the developers which could lead to tracking and potentially a Browser-In-The-Middle attack. Finally, a comparison with results revealed that hardware or emulation testing yields similar results and that this research paper identifies more information retained in RAM and less on disk, proving that contrary to browser claims, browsers on android retain information even when they say they do not.

***Keywords*— forensics, browsers, android, mobile**

1 Introduction

1.1 Background

People use the internet to access a variety of services and information, including banking, social media and political advocacy. They access the internet using web browsers which are available on desktop and mobile devices. These web browsers save browsing history and cookies (Kerschbaumer et al. 2018). Browsing history is saved to keep a record of sites visited, making it easier to come back to them when needed. Cookies were originally designed to maintains a user session to make them login, but have been also used to track users over the internet. This information reveals what a person does while using a browser, which is not always wanted. For example, a victim of domestic abuse may not want to advertise that they visited websites trying to help them.

Browsers created a private browsing mode that claim that they do not store this information on disk (Kerschbaumer et al. 2018). As that is an important claim, it needs to be researched. There has been a number of research papers looking into desktop private modes for browsers, including Horsman et al. (2019), Hughes et al. (2021), Jadoon et al. (2019) and Muir et al. (2019) whose discoveries concluded that every browser retains at least some browsing information on disk and RAM. No research was complete to see if the same applies to mobile browsers to the best of my knowledge. Additionally, previous research was completed only through emulation, casting a doubt at the accuracy of the

research as discovered by Cassel et al. (2022) who argued there are differences between using hardware and emulating. Furthermore, previous research did not look into the network connections happening on private modes, which could enable browser developers to launch a browser in the middle attack as described by Tommasi et al. (2022). Finally, I discovered from Pugliese et al. (2020) and Smith & Guzik (2022) that people know about and use browser addons, which could unknowingly save information on private mode usage on the browser through either unexpected browser updates (Franken et al. 2019) or through storing information through DOM elements (Picazo-Sanchez et al. 2020), as addons have elevated privileges.

1.2 Justification and Contribution

As I have discovered that not enough research was complete on mobile android browser private modes, I propose following a similar methodology to other desktop papers to uncover what, if any, information gets retained on mobile browsers on disk and in RAM. Additionally, I use the research created by others and compare it to my own discoveries, which enables me to check if results mirror cross platform. For that reason, I check the browsers Brave, Chrome, Firefox and Tor, as these were tested in previous papers. Additionally, I realised that no one analysed the network connections that occur in private browsing mode, which is also researched in this paper. Finally, I check if browser addons affect private modes results.

I also discovered that previous papers only attempt to do their research through emulation. For this reason my paper does all of the above experiments both on emulation and hardware. This will enable me to compare results and find out if emulation is an effective method of research as compared to using real hardware, allowing me to verify if it is likely that previous research would yield the same results if attempted on real hardware.

The research question is - "What information can be learned from a browser private window?" that is divided into the following subheadings:

- *Examine what information is retained on a mobile android device after exiting private browsing?*
- *Compare discoveries completed by this research with those of Muir et al. (2019) and Hughes et al. (2021).*
- *Examine if mobile android browsers phone back to their developers.*
- *Examine if browser addons affect private browsing windows.*
- *Examine if the results differ when emulating or on real hardware.*

1.3 Organisation of the paper

This paper discusses the literature review that was completed for this paper in section 2. The research methodology describes how we conduct our experiments in section 3. Section 4 covers the design specification that is needed to perform the implementation. Section 5 introduces the results that were a result of following the methodology, with section 6 providing an evaluation of those results and comparing them to other papers. Finally, the conclusion and future work is discussed in section 7.

2 Literature Review

2.1 Forensic Analysis of Browsers Private Modes

Using web browsers normally saves usage information on a persons computer, including cookies and browsing history (Kerschbaumer et al. 2018). This may be undesirable when a person wants to hide the fact that they used a browser to access the internet. Modern browsers include a private browsing mode, which allows a person to access the internet without saving any information about that fact on device (Kerschbaumer et al. 2018). Mobile forensics is a branch of digital forensics which itself is a branch of forensic science (Lee & Luallen 2018). Lee & Luallen (2018) describe forensic science as a method of gathering information about a device and examining its contents, which are used to answer a question. From my research, there were a number of forensic analyses of browsers available:

Horsman et al. (2019) introduced a methodology for reviewing browser private modes effectiveness of not retaining information browsed through information stored on disk and RAM. Their work discovered that out of 30 selected browsers, only 22 were available publicly, had a private browsing mode and could run on Windows 10. Horsman et al. (2019) research contributes that while a browser may have private mode and claim it does not store information, the truth may be different. The research however does not test on real hardware and emulates a Windows environment inside a VirtualBox image. Horsman et al. (2019) scope did not cover mobile browsers or other operating systems, highlighting a gap in research for other browser platforms and testing on real hardware.

Hughes et al. (2021) paper developed a similar methodology to Horsman et al. (2019) of forensically examining browser private windows. Hughes et al. (2021) has tested browsers that were reviewed in Horsman et al. (2019) paper, allowing me to compare results and evaluate if there was a change in the 2 years between the papers. Similar to Horsman et al. (2019), Hughes et al. (2021) performed their research emulating a Windows 10 Pro VMWare machine and did not test mobile browsers. Cassel et al. (2022) spoke about browser web tracking, performing a study using real android and desktop browsers. They have also evaluated if there are differences with using emulation software against real hardware and have concluded that there are various differences between them. Their discovery between hardware and emulation is significant as using emulation may have affected previous research completed in this area.

Neither Horsman et al. (2019) nor Hughes et al. (2021) papers have covered a forensic examination of the Tor Browser. The Tor Browser is a modified Firefox client which uses the Onion Router to provide anonymous online communication if used correctly by not logging into websites where a persons identify is known before a tor connection (Saleem et al. 2022). By design, Tor only runs through a private window, deleting user information upon termination (Jadoon et al. 2019). Jadoon et al. (2019) conducted such a forensic examination on the Tor Browser and discovered that Tor does leak information, especially in RAM. Jadoon et al. (2019) paper also covers Tor browser through emulation only and does not perform an analysis on other operating systems other than Windows 8.1. Muir et al. (2019) experiments on Tor have also concluded that Tor saves information to disk.

2.2 Understanding the Types of Fingerprinting Possible in Browsers

There was a 3 year study that was conducted online by Pugliese et al. (2020) on browsing fingerprinting that involved 1,300 people, allowing researchers to understand the trace-

ability of people based on their browser. Browser fingerprinting is described by Laperdrix et al. (2020) as the process of combining information known about a browser to form a unique identification for a device, giving the ability to track a person over the internet. Pugliese et al. (2020) research found that awareness of fingerprinting did not alter most users behaviour but it did show that 38% of respondents used browser addons such as uBlock Origin to block fingerprinting scripts. This revealed to me that a relatively large number of people are willing to use browser addons to protect themselves such as one explored by Ajay & Guptha (2022) who managed to create an extension to mask device attributes before they are intercepted by websites, reducing the chance of being successfully fingerprinted.

To contrast Pugliese et al. (2020) general research that aimed to question the population from various backgrounds and skills, Smith & Guzik (2022) performed interviews with developers invested in privacy and security who develop browser addons. The paper and its participants argued that privacy addons are a necessary tool in order to counter state surveillance as revealed by whistleblower Edward Snowden (Smith & Guzik 2022). One of the addons explored was LessPass, a password manager that generates passwords based on information given. Liang et al. (2018) explored creating such a secure password manager extension based on a URL domain and a master password to regenerate unique passwords, finding that it is practical and does not take much resources. Smith & Guzik (2022) did not discuss browser addons having the potential of retaining information on device during private browsing as discussed by Picazo-Sanchez et al. (2020) in the paragraph below.

Picazo-Sanchez et al. (2020) provided insights on how browsers addons may be hurting both user privacy and security by having the ability to store and read visited websites through DOM elements. This may enable any addon maintainer knowingly or not, to view and store elements of visited websites, even in private browsing modes, breaking the trust model where it is assumed no history is saved. Ferguson (2019) explains how browser addons are quickly being adopted for research for library services, but just as Picazo-Sanchez et al. (2020) states, they are concerned for the security of their patron information, as addons may be collecting usage data without knowledge. As new exploits in browsers are discovered and new features are added, browsers alter the way they and addons work; Franken et al. (2019) discovered how browser updates may have unforeseen behaviour, which could alter the way an addon works and result in storing information from private browsing, which it was not designed to do. Franken et al. (2019) have not tested addon behaviour in incognito browsing modes or on mobile devices.

Mazel et al. (2019) is another paper that analyses different techniques to protect browser privacy by using different addons and changing browser settings. They revealed that "hardening" browser settings and installing addons such as NoScript is effective in protecting user privacy against fingerprinting. For future work, they suggest that they wish to examine more browsers because they have only completed their study on Firefox. The paper also only looked at the desktop version of Firefox, neglecting android. Mitchell & Al-Fannah (2020) is a similar paper to Mazel et al. (2019) that recommends that all privacy preserving settings that exist in a browser should be turned on when possible to avoid leaking unnecessary information.

Modern applications on any device have the capability to send statistics or telemetry on its usage back to its developers. Leith (2021) submitted a paper that studied this phenomenon on desktop and mobile browsers. It revealed Chrome, Edge, Firefox and Safari did send telemetry back to its developers, while Brave did not. Leith (2021)

however, did not conduct experiments within browser private modes.

Sending telemetry is potentially dangerous as it may allow browser developers to perform targeted attacks through malice or through a court order. One such attack that can be performed is browser-in-the-middle (BitM) attack as proposed by Tommasi et al. (2022). This attacks allows an adversary to observe what a specific user is doing in a browser and even change what they see. The research by Tommasi et al. (2022) relies on a victim being phished, clicking on a link expecting to go to a website. The victim is redirected to a website controlled by the adversary that streams it to the victim. The victim logs in and is able to communicate with others, with the adversary monitoring everything.

2.3 Research Niche

To summarise the literature review, Horsman et al. (2019) and Hughes et al. (2021) performed forensic analysis of desktop browsing private modes. They covered what information is retained on device and in RAM but did not cover mobile devices. They also have not covered the Tor browser but Jadoon et al. (2019) has, but also only on desktop. Furthermore, the above research papers only conducted experimentation inside virtual machines. We know from Cassel et al. (2022) that testing on real hardware and in virtualised environments may sometimes yield different results. In this research, a similar methodology to (Muir et al. 2019) and Hughes et al. (2021) to discover if android browser private modes retain information after terminating a session. Firefox, Chrome, Brave and the Tor browsers will be tested which will allow to compare this papers results to the other research papers. These specific browsers are chosen because they occur in the cited research papers, allowing a comparison. Emulation and Hardware results will also be compared to identify if previous research which was all done through emulation can be trusted.

Horsman et al. (2019), Hughes et al. (2021), and Muir et al. (2019) did not look into the connections that were being made while using private modes. Leith (2021) looked into only normal browsing modes on desktops and mobile. In this research, I will look at the connections being made from using private windows and compare them to results from Leith (2021) to see if normal browsing and private browsing modes mirror.

Pugliese et al. (2020), and Smith & Guzik (2022) looked into how using browser addons may be useful for privacy and security. It is therefore expected that individuals will use addons for that reason, which is also shown by the vast amount of papers dedicated to addons as seen in Ajay & Guptha (2022), Liang et al. (2018), Mazel et al. (2019) and Mitchell & Al-Fannah (2020). Picazo-Sanchez et al. (2020) and Ferguson (2019) warn that browser addons may be storing information on behaviour without user knowledge, while Franken et al. (2019) warns that browser updates may have unintended consequences for addons. These updates may make a browser update cause addons to store information from private windows unintentionally. As these papers did not look into using addons with private windows, I will examine if using select addons retains information on device and compare my results to private browsing without addons.

The above points allow me to formulate the following research questions that I will contribute to answer:

- Examine what information is retained on a mobile android device after exiting private browsing?

- Compare discoveries completed by our research with those of Muir et al. (2019) and Hughes et al. (2021).
- Examine if mobile android browsers phone back to their developers.
- Examine if browser addons affect private browsing windows.
- Examine if the results differ when emulating or on real hardware.

3 Research Methodology

3.1 Research Method

Both Hughes et al. (2021), Horsman et al. (2019) and Muir et al. (2019) provided research into private browsing modes for desktop browsers using virtualisation. Their research is similar, allowing to adapt some of their methodologies to this research for android browsers. The reason behind using a similar methodology is to later compare this papers android results with their desktop results.

3.1.1 Base Setup

There are a number of constants that are present throughout the methodology, such as android studio emulation and Samsung hardware setup. To not repeat points, all assumptions and base setup settings are listed here. Unless otherwise stated, this is the setup. **The versions of the tools used are stated** in Table 16.

The browser versions were downloaded from APKMirror.com as it is not possible to rollback versions on the Google Play Store. This is important as it was needed to select the most recent versions of each browser and continue using that throughout the weeks incase a browser was updated. Two sets of browser versions were downloaded; x86_64 and arm64-v8a. This is because the emulator works on a x86_64 architecture while the Samsung phone on arm64-v8a.

Testing is conducted in two ways; through emulation and hardware. Virtualisation software used by previous researchers was not applicable as they simulate desktop operating systems, therefore I experimented with different android emulators such as Genymotion, NoxPlayer and Android Studio. It was decided that Android Studio was to be used because it is a standard android development environment with an emulator used for testing android applications. Creating the Android emulator, the Pixel 6 Pro screen was used with a system image of Tiramisu (Android 13, API33) with default settings except the boot option was set for 'Cold Boot', 6GB of RAM, 6GB VM Heap, 20GB internal storage and no SDCard storage. Snapshots were utilised to enable rollback to a clean state (before the installation of a browser) to speedup the process as otherwise the emulator would have to be reset each time, costing time. Hughes et al. (2021) have also used snapshot functionality in Virtualbox to speed up their research.

The mobile phone is rooted using Odin and Magisk and is reset to factory settings for each experiment. The hardware that will be used for the research is a Samsung Galaxy S20 5G model SM-G981x because that is what was available.

All experiments were completed twice to make sure that the results are reproducible.

3.1.2 What data leakage occurs when exiting a browsers private window on an android device?

For each browser experiment, the steps outlined below are followed. These steps were also performed on normal browsing to check what information was retained normally to compare the results with private browsing.

1. Turn on a clean android environment through reset or snapshot along with Frida.
2. Install and start a browser.
3. Perform tasks from Table 1.
4. Export browser data which is located in /data/data/ directory and perform RAM capture with Fridump.

Table 1: List of tasks to follow during browser testing.

Task	Action
1	Go to https://www.ncirl.ie/ and close the tab after a full load.
2	From two different tabs - go to https://www.youtube.com/ and https://www.eff.org/.
3	Open another tab and go to https://www.google.com and login.
4	From the google tab, go to https://signal.org/android/apk/ and download the Signal APK
5	Using the default search engine, query *'Wikipedia and Tutanota'* via the URL bar.
6	Using the application manager, terminate the browser and open it again.

To acquire data from the android device, we follow a similar *live acquisition* approach to Hughes et al. (2021) research: Hughes et al. (2021) performed their analysis in two parts; by exporting the disk and by capturing RAM:

1. **Exporting the disk:** Using my host machine, I connected the Android device to ADB and used 'adb pull' to pull the browser data from the /data/data/ directory. For example, for Brave, I export the com.brave.browser folder from /data/data/.
2. **Capturing RAM:** Fridump with Frida is used to dump RAM. The Linux Memory Extractor (LiME) was attempted to be used to perform a RAM capture as it is the more obvious choice but that was impossible as LiME requires to use a custom kernel module to be installed which Samsung did not release for our particular phone model. To use Fridump, the Frida server was transferred to /data/local/tmp, given 'chmod 755' permissions to run and activated. When an experiment is finished, I ran Fridump to dump RAM by using the 'python fridump.py -U -s *Browser Service*' command.

The disk and RAM is then zipped and hashed with SHA256 for backup. The windows strings command to search the extracted browser folder for key words such as 'NCIRL', 'Signal' and 'Tutanota' is used for evaluation. For RAM, fridump performs the string command automatically to create a file. In that file, I used the Notepad find function to find keywords such as 'NCIRL'. I then create a table to list all of my findings in section 5.

3.1.3 Compare discoveries completed by our research with those of Muir et al. (2019) and Hughes et al. (2021).

This experiment involves comparing my results from section 3.1.2 with those of Muir et al. (2019) and Hughes et al. (2021). Those papers followed a similar methodology to this paper, allowing a comparison to take place.

3.1.4 Examine if mobile android browsers phone back to their developers

This research was spearheaded by Leith (2021) who devised a similar methodology to the one this paper uses. In order to observe the communication that occurs between the browser and the internet, MITMProxy was used. It allows to man-in-the-middle HTTPS connections that occur within the browser. MITMProxy required me to install a CA certificate on the android environment and proxy my android connection to the host machine, allowing to see HTTPS connections from the host.

To examine the browser, the paper adopted a similar approach to Leith (2021) where the steps in Table 2 are followed.

Table 2: List of tasks to follow during browser testing.

Task	Action
1	MITMProxy is online on the host machine and browser installed on android.
2	The browser is started in a private window.
3	In the address bar, paste https://www.ncirl.ie/.
4	Terminate the browser by using the application manager.
5	Open the browser in a private window and leave it for 1 hour.
6	Type https://www.ncirl.ie/ manually, ensuring consistent typing speed.

After each experiment ends, export the MITMProxy data and examine the connections through MITMWeb. The results are then analysed by looking each HTTP request and compared to Leith (2021) findings, contributing to them as he did not cover private browsing modes.

3.1.5 Examine if browser addons affect private browsing windows.

Mazel et al. (2019) discovered that web privacy protections through browser addons may improve protection on desktop devices, this research work examines if browser addons affect private browsing windows on android. This research performs the same methodology on browsers as in section 3.1.2 using UBlock Origin and Dark Reader addons. The results are then compared to those without addons to understand if results are different.

3.1.6 Examine if the results differ when emulating or on real hardware.

Cassel et al. (2022) explored that there are sometimes differences between using emulation and real hardware. Each examination from the sections above is completed through both emulation and real hardware to compare results and determine if the results are significant enough to question the validity of previous papers as previous research papers in this area have used virtual machines to perform experiments on browsers.

4 Design Specification

The implementation relies on a number of factors. It is complete using a TUF GAMING FX504GE_FX80GE laptop fitted with an Intel Core i7-8750H @ 2.20GHz CPU, 16GB of RAM (2 x 8GB 1Rx8 PC4-2666v) and a 240GB SSD (BX500 2.5"). This laptop is the host machine and is used to collect data from the android phones, either through virtualising an android device or by using android studio with ADB to connect to the physical phone. The phone is a Galaxy S20 5G SM-G981x.

Figure 1: Setup used for emulating android for private mode experiments (with and without addons)

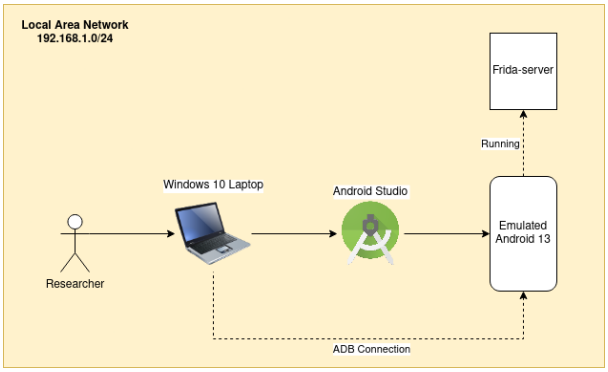


Figure 1 shows the architecture of the implementation to perform tests for section 3.1.2 and 3.1.5. The researcher controls the Windows 10 host machine which runs Android Studio. Android Studio is emulating an Android 13 environment from where experiments from the methodology take place.

Figure 2: Setup used for android hardware for private mode experiments (with and without addons)

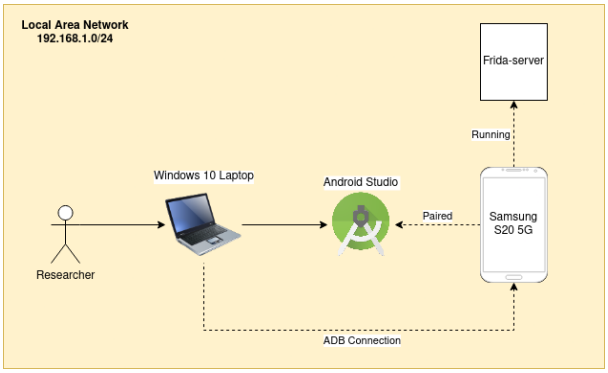


Figure 2 follows a similar design to Figure 1 but the hardware phone is paired to Android Studio via Wifi in order to connect with ADB. On both instances, frida-server is running in order to run fridump to dump RAM. As the RAM is dumped, the RAM dump along with the browser storage is transferred to the Windows host in order to be evaluated after experiments are over.

Figure 3: Setup used for emulating android for private mode network experiments.

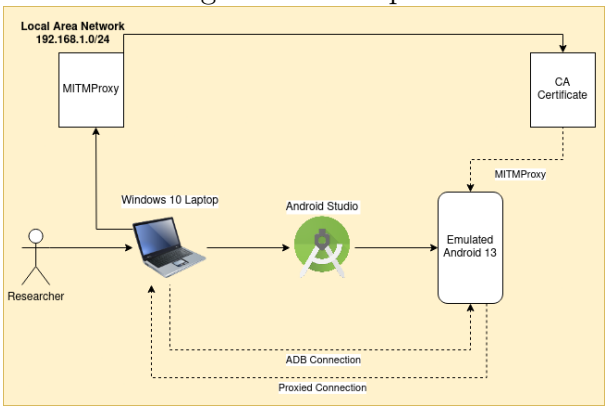
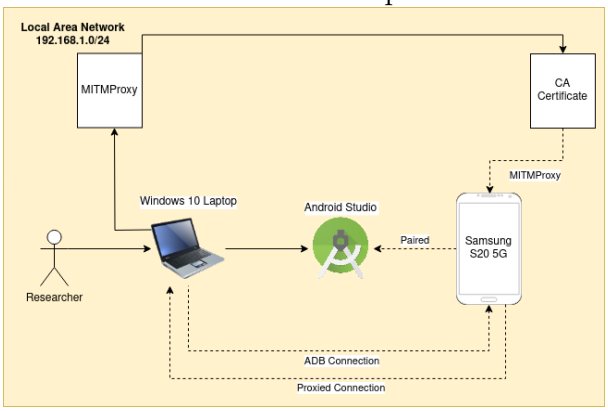


Figure 3 shows the architeture used for the implementation for section 3.1.4. As before, a Windows host is used but this time MITMProxy is running on it during the experiment. A CA Certificate is installed on the Android device, with the internet connection proxied to the Windows host to observe HTTP requests. At the end of the experiment, MITMProxy traffic is saved to the host machine. Figure 4 has the same setup but the phone is paired to Android Studio rather than emulated.

Figure 4: Setup used for android hardware for private mode network experiments.



5 Implementation

5.1 What information is retained from browser private modes experiment?

Table 3 through 10 shows the results for what information on disk and RAM was revealed from privacy modes upon doing the experiments outlined in the methodology. It was revealed that Chrome could not be installed on Android Studio as Android on emulation does not want to install outdated versions of chrome (this was not a problem on hardware). For this reason, Chrome on emulation is marked as 'N/A'. The Tor Browser is also marked as 'N/A' on normal browsing modes experiment as it does not have a normal browsing mode.

The columns represent the search term that was searched for in browser folders using the Powershell Get-String command. For example, I searched for "NCIRL" to search for instances where NCIRL occurred using "Get-ChildItem -Recurse | Select-String -Pattern "ncirl" -List. Fridump from the RAM dump produces its own string file for which Notepad's find function was used to find key terms.

5.1.1 Normal Browsing Mode

Table 3: **Disk storage output** testing **normal browsing** using **emulation**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	Yes	Yes	Yes	Yes	Yes
Brave Test 2	Yes	Yes	Yes	Yes	Yes
Chrome Test 1	N/A	N/A	N/A	N/A	N/A
Chrome Test 2	N/A	N/A	N/A	N/A	N/A
Firefox Test 1	Yes	Yes	No	Yes	Yes
Firefox Test 2	Yes	Yes	No	Yes	Yes
Tor Test 1	N/A	N/A	N/A	N/A	N/A
Tor Test 2	N/A	N/A	N/A	N/A	N/A

Table 4: **RAM dump** output testing **normal browsing** using **emulation**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	Yes	Yes	Yes	Yes	Yes
Brave Test 2	Yes	Yes	Yes	Yes	Yes
Chrome Test 1	N/A	N/A	N/A	N/A	N/A
Chrome Test 2	N/A	N/A	N/A	N/A	N/A
Firefox Test 1	Yes	Yes	No	Yes	Yes
Firefox Test 2	Yes	Yes	No	Yes	Yes
Tor Test 1	N/A	N/A	N/A	N/A	N/A
Tor Test 2	N/A	N/A	N/A	N/A	N/A

Table 5: **Disk storage output** testing **normal browsing** using **hardware**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	Yes	Yes	Yes	Yes	Yes
Brave Test 2	Yes	Yes	Yes	Yes	Yes
Chrome Test 1	Yes	Yes	Yes	Yes	Yes
Chrome Test 2	Yes	Yes	Yes	Yes	Yes
Firefox Test 1	Yes	Yes	No	Yes	Yes
Firefox Test 2	Yes	Yes	No	Yes	Yes
Tor Test 1	N/A	N/A	N/A	N/A	N/A
Tor Test 2	N/A	N/A	N/A	N/A	N/A

Table 6: **RAM dump** output testing **normal browsing** using **hardware**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	Yes	Yes	Yes	Yes	Yes
Brave Test 2	Yes	Yes	Yes	Yes	Yes
Chrome Test 1	Yes	Yes	Yes	Yes	Yes
Chrome Test 2	Yes	Yes	Yes	Yes	Yes
Firefox Test 1	Yes	Yes	No	Yes	Yes
Firefox Test 2	Yes	Yes	Yes	Yes	Yes
Tor Test 1	N/A	N/A	N/A	N/A	N/A
Tor Test 2	N/A	N/A	N/A	N/A	N/A

5.1.2 Private Browsing Mode

Table 7: **Disk storage output** testing **private browsing** using **emulation**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	No	No	No	No	Yes
Brave Test 2	No	No	No	No	Yes
Chrome Test 1	N/A	N/A	N/A	N/A	N/A
Chrome Test 2	N/A	N/A	N/A	N/A	N/A
Firefox Test 1	No	No	No	No	No
Firefox Test 2	No	No	No	No	No
Tor Test 1	No	No	No	No	No
Tor Test 2	No	No	No	No	No

Table 8: **RAM dump** output testing **private browsing** using **emulation**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	No	Yes	Yes	No	Yes
Brave Test 2	No	Yes	No	No	Yes
Chrome Test 1	N/A	N/A	N/A	N/A	N/A
Chrome Test 2	N/A	N/A	N/A	N/A	N/A
Firefox Test 1	Yes	Yes	Yes+partial password	Yes	Yes
Firefox Test 2	Yes	Yes	Yes+partial password	Yes	Yes
Tor Test 1	No	No	No	No	No
Tor Test 2	No	No	No	No	No

Table 9: **Disk storage output** testing **private browsing** using **hardware**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	No	No	No	No	Yes
Brave Test 2	No	No	No	No	Yes
Chrome Test 1	No	No	No	No	No
Chrome Test 2	No	No	No	No	No
Firefox Test 1	No	No	No	No	No
Firefox Test 2	No	No	No	No	No
Tor Test 1	No	No	No	No	No
Tor Test 2	No	No	No	No	No

Table 10: **RAM dump** output testing **private browsing** using **hardware**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	Yes	Yes	Yes	Yes	Yes
Brave Test 2	Yes	Yes	Yes	Yes	Yes
Chrome Test 1	No	Yes	No	No	Yes
Chrome Test 2	No	Yes	Yes	No	Yes
Firefox Test 1	Yes	Yes	Yes+partial password (ictrashpillow)	Yes	Yes
Firefox Test 2	Yes	Yes	Yes+partial password (ictrashpillow)	Yes	Yes
Tor Test 1	No	No	No	No	No
Tor Test 2	No	No	No	No	No

5.2 Browser Private Window HTTP Requests

As before, Chrome was not possible to install in emulation. It was also not possible to view HTTP requests from Tor which is therefore marked 'N/A'. Each number in Table 11 represents a HTTP request sent to a developer which could be used to form a profile on a users private browsing usage. These include but are not limited to android version, browser version, time, screen size, addons installed, pages viewed.

Table 11: Private Browsing HTTP requests sent to developers.

Browser Tested	Emulation	Hardware
Brave Test 1	113	123
Brave Test 2	106	122
Chrome Test 1	N/A	50
Chrome Test 2	N/A	30
Firefox Test 1	27	25
Firefox Test 2	25	41
Tor Test 1	N/A	N/A
Tor Test 2	N/A	N/A

5.3 Browser Private Window with Addons

Table 12 to 15 show what information is saved when using private mode using the same search pattern from section 5.1. Addons were only possible to be installed on Firefox and

Tor which is the reason other browsers are marked as 'N/A'.

Table 12: **Disk output** with addons testing **private browsing** using **emulation**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	N/A	N/A	N/A	N/A	N/A
Brave Test 2	N/A	N/A	N/A	N/A	N/A
Chrome Test 1	N/A	N/A	N/A	N/A	N/A
Chrome Test 2	N/A	N/A	N/A	N/A	N/A
Firefox Test 1	Yes	Yes	No	No	No
Firefox Test 2	Yes	Yes	No	No	No
Tor Test 1	No	No	No	No	No
Tor Test 2	No	No	No	No	No

Table 13: **RAM dump** with addons output testing **private browsing** using **emulation**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	N/A	N/A	N/A	N/A	N/A
Brave Test 2	N/A	N/A	N/A	N/A	N/A
Chrome Test 1	N/A	N/A	N/A	N/A	N/A
Chrome Test 2	N/A	N/A	N/A	N/A	N/A
Firefox Test 1	Yes	Yes	Yes+partial password	Yes	Yes
Firefox Test 2	Yes	Yes	Yes+partial password	Yes	Yes
Tor Test 1	No	No	No	No	No
Tor Test 2	No	No	No	No	No

Table 14: **Disk output** with addons testing **private browsing** using **hardware**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	N/A	N/A	N/A	N/A	N/A
Brave Test 2	N/A	N/A	N/A	N/A	N/A
Chrome Test 1	N/A	N/A	N/A	N/A	N/A
Chrome Test 2	N/A	N/A	N/A	N/A	N/A
Firefox Test 1	Yes	Yes	No	No	No
Firefox Test 2	Yes	Yes	No	No	No
Tor Test 1	No	No	No	No	No
Tor Test 2	No	No	No	No	No

Table 15: **RAM dump** with addons output testing **private browsing** using **hardware**.

Browser Tested	NCIRL	EFF	x21244545@gmail.com	Signal	Tutanota
Brave Test 1	N/A	N/A	N/A	N/A	N/A
Brave Test 2	N/A	N/A	N/A	N/A	N/A
Chrome Test 1	N/A	N/A	N/A	N/A	N/A
Chrome Test 2	N/A	N/A	N/A	N/A	N/A
Firefox Test 1	Yes	Yes	Yes+FULL PASSWORD	Yes	Yes
Firefox Test 2	Yes	Yes	Yes+partial password (ictrashpillow)	Yes	Yes
Tor Test 1	No	No	No	No	No
Tor Test 2	No	No	No	No	No

5.4 Research Resources

Table 16 shows the tools used.

Table 16: Tool name, description and version used for experiments.

Tool Name	Description	Version
Chrome	Browser used for examination	114.0.5735.196-573519627
Brave	Browser used for examination	1.52.129-415212927
Firefox	Browser used for examination	115.0-2015958819
Tor Browser	Browser used for examination	102.2.1-Release_(12.5)-2015960783
Hardware	Hardware Device	Galaxy S20 5G SM-G981x
Frida	Toolkit used for RAM dump	16.0.19
Fridump	Performs RAM dump	0.1
Android	Android version	13 (API 33)
ADB	For communication with android	1.0.41
MITMProxy	MITM Service	9.0
Strings	Finds readable strings in files	2.54 (SysInternals version)
Android Studio	Used for emulating an android device	Flamingo — 2022.2.1 Patch 2
Windows	Operating system used.	10.0.19045
Odin	Used to flash Samsung phone	3.13.1
Magisk	Used to root the Samsung device.	d390ca2f (26104)
uBlock Origin	Addon used to test private browsing	1.51.0
Dark Reader	Addon used to test private browsing	4.9.64

6 Evaluation

6.1 Private Browsing Findings

6.1.1 Brave

Brave behaved as is expected from normal browsing through both emulation and hardware experiments: all query information from the two experiments was saved as can be observed in Table 3 through 6. In private browsing mode tables 7 through 10, no query information was retained on disk except tutanota. This may alert an adversary performing forensic analysis about a person using Tutanota for email communication,

causing the victim stress. When examining the RAM dump, the results show a different story, with the browser revealing partial queried information through emulation and **all** queried information through hardware. This has massive implications if an adversary is able to receive the device immediately from the victim to perform an analysis. It also has interesting implications for discussing emulation and hardware, covered in section 6.4.

6.1.2 Chrome

Unfortunately, Chrome was only possible to install from hardware and not emulation, giving only one source of truth. As expected, normal browsing has saved all queried information on device just like Brave. Private browsing on Chrome revealed that no information was retained on disk after the browser was terminated. The RAM dump however did reveal that EFF, Tutanota and the email was visited. The email was identified in only one test while the others queries were present on both.

6.1.3 Firefox

Firefox revealed interesting results from normal browsing. Table 3 through 6 experiments did not retain the fact that the browser logged into a gmail account both on hardware or emulation on disk **or** ram. It did however, retain everything else on disk, with experiment two additionally revealing the email in RAM as seen on table 6. This finding is interesting as it may reveal some sort of protection regarding saving login information by default. Private browsing in tables 7 through 10 saved no query information from the experiments on disk. It was possible to recover all queried information from the RAM dumps however. It was also possible to recover half of the gmail password used - "ictrashpillow". No other browser had this type of information leak. This could allow an adversary who confiscated a victims phone to possibly bruteforce a users password, as the length of it was halved. What is also strange is that private browsing mode leaked the password and email while normal mode did not.

6.1.4 Tor

As Tor only has one default mode of browsing - private mode - it was not possible to examine if it retains information in browsing normally. As it is designed with security and privacy in mind, Tor is *expected* to not retain anything both on disk and RAM when terminated. The experiments in tables 7 through 10 revealed that Tor was that only browser both on disk and RAM to not retain browsing history.

6.2 Private Browsing with Addons Findings

6.2.1 Firefox

Results from Table 12 and 14 found that Firefox kept some information on disk both on emulation and hardware - the NCIRL and EFF queries. This contrasts to the results revealed in private windows *without* addons where this information was not recovered. The RAM results revealed mostly the same information except for hardware results seen in figure 15 where it was possible to recover the **full** password used to login to Gmail, which was not possible on Firefox *without* addons. As the addon tests were performed using the same methodology as without addons which did not leak any data on disk, it suggests that Firefox addons do infact alter the way information is stored in private

windows. This finding is significant for both academic and practical purposes as users of browsers need to be conscious of the addons they install, as they may leak undesirable information on private windows. For academic purposes, future papers will need to incorporate browser addon tests as testing private windows alone is not enough as we learned that many people are interested or are using addons from Pugliese et al. (2020) and Smith & Guzik (2022).

6.2.2 Tor

Results from table 12 to 15 revealed that no information was stored with Tor while using addons. These results mirror those from not using browser addons discussed in the previous sections. As Tor is based on Firefox, these results may be a result of Tor hardening the browser against fingerprinting.

6.3 Private Browsing Network Connections

Table 11 shows the number of connections that have occurred in the hour of running the browser experiments. These connections represent requests sent to the developer which may uniquely identify a user using either unique identifiers or a combination of statistics about a user (android version, browser version, time, screen size, addons installed, pages viewed) which may enable developers to perform a browser-in-the-middle attack as described by Tommasi et al. (2022). The research revealed that Brave utilised such requests the most, then Chrome and lastly, Firefox the least. From examining those requests, while Brave sent the most requests, they also carried less information on each request than the other browsers, likely hindering their ability to accurately fingerprint a unique browser over time. The other browsers aggregated more information on their requests, making it easier to fingerprint unique browsers, enabling browser developers to make a targeted attack on an unsuspecting victim. Collecting usage information on browser usage is not inherently malicious, but could indicate that browser developers do not care about user privacy, as the experiments were ran on browser private windows, which users assume means they are private from those queries being analysed, even if these browsers do not claim they do not see what users are doing, except Tor.

6.4 Differences in Emulation and Hardware?

Table 17: Findings of Browser Private Windows

	Without Addons				With Addons			
	Emulated		Hardware		Emulated		Hardware	
	Disk	RAM	Disk	RAM	Disk	RAM	Disk	RAM
Brave	2/10	5/10	2/10	10/10	N/A	N/A	N/A	N/A
Chrome	N/A	N/A	0/10	5/10	N/A	N/A	N/A	N/A
Firefox	0/10	10/10	0/10	10/10	4/10	10/10	4/10	10/10
Tor	0/10	0/10	0/10	0/10	0/10	0/10	0/10	0/10

Table 17 show the differences between emulated and hardware results. Each browser experiment has 5 total queries that are possibly recoverable, with each browser experiment being completed 2 times, making the sum of possible recoverable artefacts 10.

It was not possible to compare Chrome on emulation as always because it was not possible to install on the emulator, however it was possible to compare Brave, Tor and Firefox. Tor and Firefox results' are identical when comparing their emulated and hardware results. Brave results are not identical on RAM but are on disk.

The table also shows that the same results are yielded for emulation and hardware when using addons, the same as without addons, but only for Firefox and Tor because the other browsers do not support addons.

As RAM is volatile by nature and would require an adversary to root a device before dumping its memory which requires device restarts, thus removing the contents stored, there is sufficient evidence in the results to believe that emulation is a sufficient medium to conduct browser experiments and does not significantly alter results. This is in contrast to Cassel et al. (2022) research on how emulation and hardware sometimes produces different results, indicating that the desktop browser research from Hughes et al. (2021), Horsman et al. (2019) and Muir et al. (2019) would likely be identical when complete using hardware.

Regarding network connections that occur when using private modes on browsers between the developer and browser, Table 11 show that hardware sends more information on average than emulation. This was hard to measure as Tor was not possible to setup with MITMProxy and Chrome did not work on emulation, meaning that the remaining browsers (Brave, Firefox) sending more requests on hardware may be coincidental as not enough samples can be compared.

6.5 Comparing results of other papers

Table 18 and 19 shows a comparison table comparing mobile results to Hughes et al. (2021)[1] and Muir et al. (2019) to determine if mobile and desktop browsers behave in the same way. As it was determined above that emulation and hardware operate in an similar fashion without loss to results, hardware results will be used to compare this papers work against others. Hardware results were chosen because chrome on emulation was not possible.

Table 18: Findings of Browser Private Windows without Addons

	This Paper (Hardware Results)		(Hughes et al. 2021)		(Muir et al. 2019)	
	Disk	RAM	Disk	RAM	Disk	RAM
Brave	2/10 (20%)	10/10 (100%)	6/7 (85.71%)	5/7 (71.43%)	N/A	N/A
Chrome	0/10 (0%)	5/10 (50%)	6/7 (85.71%)	0/7 (0%)	N/A	N/A
Firefox	0/10 (0%)	10/10 (100%)	5/7 (71.43%)	4/7 (57.14%)	N/A	N/A
Tor	0/10 (0%)	0/10 (0%)	N/A	N/A	Found leaks	Not found

This papers results show significant differences to Hughes et al. (2021) with Hughes et al. (2021) being able to recover a more significant[2] amount of information from the browsers on disk for Brave, Chrome and Firefox. This paper recovered more information from RAM than previous research. As Hughes et al. (2021) performs their experiments on a desktop windows machine and this paper on Android, the results signal that information is stored differently between the same browsers on different platforms. Changes could

[1]Results captured from Table 5 and Table 7 'Strings' Tool in their research.

[2]Values greater than 28.57% (2/7) are considered significant.

Table 19: Findings of Browser Private Windows with Addons

	This Paper (Hardware Results)		(Hughes et al. 2021)		(Muir et al. 2019)	
	Disk	RAM	Disk	RAM	Disk	RAM
Brave	N/A	N/A	6/7 (85.71%)	5/7 (71.43%)	N/A	N/A
Chrome	N/A	N/A	6/7 (85.71%)	0/7 (0%)	N/A	N/A
Firefox	4/10 (40%)	10/10 (100%)	5/7 (71.43%)	4/7 (57.14%)	N/A	N/A
Tor	0/10 (0%)	0/10 (0%)	N/A	N/A	Found leaks	Not found

also be due to the fact that this papers browsers are more modern than the previous research complete.

Regarding Tor, this paper discovered no recoverable artefacts from disk or RAM in both emulation and hardware. These findings are contrary to Muir et al. (2019) findings who recovered artefacts from disk on desktop. This could suggest that the Tor browser is improving on removing traces of use from their browsers, helping people maintain their privacy and security.

Regarding network requests between browsers and its developers, this research finds that private browsing mirrors the same results discovered by Leith (2021) who tested normal browsing modes; that Chrome and Firefox use identifiers leading to fingerprinting over time with Brave not using identifiers.

6.6 Discussion

This paper completed three main pieces of work for four different browsers; a forensic analysis of private modes with addons, without addons, and HTTP connections. For the first piece of work, it was discovered that browsers still leak information in private windows even when they say they do not, with the exception of Tor which saved no artefacts on disk or RAM. Firefox has also retained half of the password that was used to login to Google using the set methodology in RAM. Emulation was not possible on Chrome due to Android not allowing the app to be downgraded from the default installed version, which may have affected the results for comparing emulation versus hardware. To improve the experiment, a way to bypass this restriction would need to be researched.

The second piece of work discovered that using browser addons may affect browser private windows. This is known because this paper discovered that using the same methodology as the the other experiment, Firefox retained some information on disk as well as stored the full Gmail password in RAM, which was not present from the experiments without addons. Tor, however, did not retain any artefacts with or without addons.

The third piece of work discovered that Brave sent the most requests back to developers when using private windows, with Chrome as second and lastly Firefox. Brave requests carried less information than Chrome and Firefox which may affect fingerprintability as there is less information to fingerprint. Chrome and Firefox sent more information inside the requests and used identifiers linking usage to a browser, unlike Brave. Due to time constraints, not enough time was spent analysing these browser HTTP requests. The most amount of time was allocated towards analysing information stored on disk and ram from private modes. It should also be noted that browser private modes do not claim that they do not send information back to developers, only that they do not store information on device.

Two other pieces of work were completed; a comparison of differences in emulation

and hardware, and a comparison of this papers result with other papers. The fourth piece discovered that emulation and hardware results mirror eachother, concluding that the previous research by Hughes et al. (2021), Horsman et al. (2019) and Muir et al. (2019) which was conducted via emulation is likely to mirror if tested using hardware.

The fifth and final piece of work compared the papers findings to those of previous research in this field. The papers research found that Hughes et al. (2021) on Brave, Chrome and Firefox were able to recover more information from their browsers than this paper, but this paper was able to recover more information from RAM than Hughes et al. (2021). On Tor, Muir et al. (2019) was able to recover artefacts from the disk while this paper was not able to recover anything from disk or RAM. These results may differ compared to Hughes et al. (2021) and Muir et al. (2019) due to the fact that browsers were updated during the times between the different research works or may be due to the fact that this paper uses a different platform (Android) to the other papers (Windows).

All of the above experiments were completed manually such as typing addresses in the URL bar or logging in to Gmail. This results in slight variations between different tests as one experiment could take more or less time than others. To improve, these tasks could be automated, ensuring that mistakes and timing errors do not happen.

7 Conclusion

This paper explored browser private modes as it had identified that research regarding android browsers was lacking in that area, with papers like Hughes et al. (2021), Muir et al. (2019) and Horsman et al. (2019) covering Windows only and only through emulation. Research was completed in five parts for Chrome, Brave, Firefox and Tor browsers;

- What information on disk and RAM is retained in private browser?
- Do results change if addons are added to the browser?
- What network requests are sent to the developers of the browsers?
- Does testing through emulation and hardware change results?
- How do results achieved compare to previous research?

It was discovered that Tor was the only browser to not retain information with and without addons in disk and RAM. Other browsers retained information about pages visited, with Firefox leaking partial passwords in private browsing and a full password when using addons. HTTP connections sent to browsers in private windows aligned with results achieved with Leith (2021) who tested normal windows only. Additionally, the results achieved through emulation and hardware mirrors eachother, indicating that there are little differences between the two mediums for testing. Finally, the results for the browsers in this paper differed, with this paper identifying more information in RAM and less on disk. To conclude, Tor is the clear favourite in this research, with it not retaining any information on disk or RAM with and without addons. Firefox is the least favourite with it leaking the Gmail password in RAM, as well as leaking a full password in RAM with addons.

7.1 Limitations

Before starting the research, I did not realise that some URLs are automatically loaded into RAM and storage. If the research was to be repeated, more unique addresses would be used rather than Youtube and Wikipedia. LiME was not able to be used during RAM dumps which is a standard way of extracting RAM from Android devices. It was not able to be used because there were no available kernal sources for the Samsung device. If the research was completed again, a compatible hardware device should be used to not rely on Fridump, which is only capable of extracting RAM of a particular running service.

7.2 Future Work

The browsers tested in this paper relied on the browser tested in previous papers, therefore, the methodology used on browsers in this paper can be adapted to test other mobile browsers, including DuckDuckGo Browser, Samsung Internet Browser, Opera browser and Mull Browser.

References

Ajay, V. L. & Guptha, A. M. (2022), 'A defense against javascript object-based fingerprinting and passive fingerprinting', *2022 International Conference on Computing, Communication, Security and Intelligent Systems (IC3SIS), Computing, Communication, Security and Intelligent Systems (IC3SIS), 2022 International Conference on* pp. 1–6. doi:10.1109/IC3SIS54991.2022.9885716.

Cassel, D., Lin, S.-C., Buraggina, A., Wang, W., Zhang, A., Bauer, L., Hsiao, H.-C., Jia, L. & Libert, T. (2022), 'Omnicrawl: Comprehensive measurement of web tracking with real desktop and mobile browsers', *Proceedings on Privacy Enhancing Technologies* **2022**(1), 227–252. doi:10.2478/popets-2022-0012.

Ferguson, C. L. (2019), 'Leaning into browser extensions', *Serials Review* **45**(1/2), 48–53. doi:10.1080/00987913.2019.1624909.

Franken, G., Goethem, T. V. & Joosen, W. (2019), 'Exposing cookie policy flaws through an extensive evaluation of browsers and their extensions', *IEEE Security & Privacy, Security & Privacy, IEEE, IEEE Secur. Privacy* **17**(4), 25–34. doi:10.1109/MSEC.2019.2909710.

Horsman, G., Findlay, B., Edwick, J., Asquith, A., Swannell, K., Fisher, D., Grieves, A., Guthrie, J., Stobbs, D. & McKain, P. (2019), 'A forensic examination of web browser privacy-modes', *Forensic Science International: Reports* **1**(-), 100036. doi:10.1016/j.fsir.2019.100036.

Hughes, K., Papadopoulos, P., Pitropakis, N., Smales, A., Ahmad, J. & Buchanan, W. J. (2021), 'Browsers' private mode: Is it what we were promised?', *Computers* **10**(12), 165. doi:10.3390/computers10120165.

Jadoon, A. K., Iqbal, W., Amjad, M. F., Afzal, H. & Bangash, Y. A. (2019), 'Forensic analysis of tor browser: A case study for privacy and anonymity on the web', *Forensic Science International* **299**, 59–73. doi:10.1016/j.forsciint.2019.03.030.

Kerschbaumer, C., Crouch, L., Ritter, T. & Vyas, T. (2018), 'Can we build a privacy-preserving web browser we all deserve?', *XRDS: Crossroads, The ACM Magazine for Students - Pseudonimity and Anonymity* **24**(4), 40–44. doi:10.1145/3220567.

Laperdrix, P., Bielova, N., Baudry, B. & Avoine, G. (2020), 'Browser fingerprinting : A survey', *CM Transactions on the Web (TWEB)* **14**(2), 1–33. doi:10.1145/3386040.

Lee, R. M. & Luallen, M. E. (2018), 'Making digital forensics: a critical part of your cyber security defenses', *Control Engineering* **61**(1). Available at: https://research.ebsco.com/linkprocessor/plink?id=e2fed71c-71f3-30ac-b601-00ef4a217b77 (Accessed: 1 August 2023).

Leith, D. J. (2021), 'Web browser privacy: What do browsers say when they phone home?', *IEEE Access* **9**, 41615–41627. doi:10.1109/ACCESS.2021.3065243.

Liang, S., Zhang, Y., Li, B., Guo, X., Jia, C. & Liu, Z. (2018), 'Secureweb: Protecting sensitive information through the web browser extension with a security token', *Tsinghua Science & Technology* **23**(5), 526–538. doi:10.26599/TST.2018.9010015.

Mazel, J., Garnier, R. & Fukuda, K. (2019), 'A comparison of web privacy protection techniques', *Computer Communications* **144**, 162–174. doi:10.1016/j.comcom.2019.04.005.

Mitchell, C. & Al-Fannah, N. M. (2020), 'Too little too late: can we control browser fingerprinting?', *Journal of Intellectual Capital* **21**(2), 165–180. doi:10.1108/JIC-04-2019-0067.

Muir, M., Leimich, P. & Buchanan, W. (2019), 'A forensic audit of the tor browser bundle', *Digital Investigation* **29**, 118–128. doi:10.1016/j.diin.2019.03.009.

Picazo-Sanchez, P., Tapiador, J. & Schneider, G. (2020), 'After you, please: browser extensions order attacks and countermeasures.', *International Journal of Information Security* **19**, 623–638. doi:10.1007/s10207-019-00481-8.

Pugliese, G., Riess, C., Gassmann, F. & Benenson, Z. (2020), 'Long-term observation on browser fingerprinting: Users' trackability and perspective.', *Proceedings on Privacy Enhancing Technologies* **2020**(2), 558–577. doi:10.2478/popets-2020-0041.

Saleem, J., Islam, R. & Kabir, M. A. (2022), 'The anonymity of the dark web: A survey', *IEEE Access* **10**, 33628–33660. doi:10.1109/ACCESS.2022.3161547.

Smith, K. L. & Guzik, E. (2022), 'Developing privacy extensions: Is it advocacy through the web browser?', *Surveillance & Society* **20**(1), 64–81. doi:10.24908/ss.v20i1.13958.

Tommasi, F., Catalano, C. & Taurino, I. (2022), 'Browser-in-the-middle (bitm) attack', *International Journal of Information Security* **21**, 179–189. doi:10.1007/s10207-021-00548-5.