University of Zurich[UZH]

BACHELOR THESIS – Communication Systems Group, Prof. Dr. Burkhard Stiller

Analysis of Unwanted Advertising Traffic: An Economic and User Experience Perspective

Lawand Muhamad
Zürich, Switzerland
Student ID: 16-729-147

Supervisor: Muriel Franco, Eder Scheid
Date of Submission: August 20, 2019

University of Zurich
Department of Informatics (IFI)
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland

ifi

Bachelor Thesis
Communication Systems Group (CSG)
Department of Informatics (IFI)
University of Zurich
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland
URL: http://www.csg.uzh.ch/

Zusammenfassung

Inhalte und Dienste, die im Internet kostenlos angeboten werden, werden in erster Linie durch Online Werbung finanziert. Das Anzeigen von Werbung stellt hierbei den Preis dar, welcher der Nutzer bezahlen muss, um den Inhalt kostenlos konsumieren zu dürfen. Viele Internetnutzer fühlen sich jedoch von der steigenden Anzahl ungewünschter und lästiger Werbung zunehmend gestört und suchen daher Möglichkeiten diese auszublenden und zu umgehen. Dies äußert sich in der stetig steigenden Beliebtheit von Adblockern auf Heimcomputern und neuerdings auch auf Smartphones.
In dieser Arbeit werden wir in einem ersten Schritt einen User Behaviour Simulator entwickeln, mit welchem es möglich ist, Datenverkehr durch das Surfen im Netz zu generieren. Dabei berechnet das Tool den Datenverbrauch sowie die benötigte Gesamtlaufzeit, um eine Liste von Webseiten zu öffnen. In einem zweiten Schritt werden mithilfe des Simulator Tools, basierend auf den meistbesuchten Webseiten der Schweiz, Messungen mit unterschiedlichen Adblocker-Tools, sowohl für Nutzer von Heimcomputern wie auch für Nutzer von Smartphones durchgeführt und näher analysiert. Zusätzlich befassen wir uns basierend auf den Resultaten mit den ökonomischen Vorteilen, welche Smartphone Nutzer mit limitierten Datenpaketen durch den geringeren Datenverbrauch haben können.
Es konnte aufgezeigt werden, dass die Nutzung eines Adblocker wie Pi-Hole zu beträchtlichen Dateneinsparungen sowie Geschwindigkeitsverbesserungen führen kann. Mobile Webseiten konnten beispielsweise über 24% schneller geladen werden und der Datenverbrauch konnte bis zu 24% reduziert werden. Auch mit den anderen Adblocker-Tools waren hierbei signifikante Verbesserungen zu beobachten.

Abstract

Content and services that are offered on the Internet for free, are financed primarily through online advertising. Viewing those advertisements is the price that the user has to pay in order to consume the content for free. However, many Internet users are increasingly disturbed by the growing number of unwanted and annoying ads and are therefore looking for ways to circumvent them. This is reflected in the ever increasing popularity of adblockers on home computers and more recently on smartphones.
In this thesis we will develop in a first step a User Behavior Simulator, with which it is possible to generate traffic by surfing the internet based on list of URLs and that is able to calculate simultaneously the data consumption as well as the total runtime to load all the web-pages. In a second step with the use of the simulator tool, based on the most visited websites in Switzerland, tests with different adblocking tools are performed, both for desktop PC users as well as smartphones users. The results of these tests are then analyzed in detail. In addition, we have a look at the economical benefits that smartphone users with limited data packages could have through the lower data consumption with adblockers.
Indeed, we were able to show that using an adblocker like Pi-Hole can lead to significant data savings as well as speed improvements not only for desktop PC user but also for smartphone users. Mobile websites, for example, were loaded 24% faster and data consumption could be reduced by up to 24%. Significant improvements were also observed with other adblocking tools.

Acknowledgments

I want to thank Professor Burkhard Stiller for giving me the opportunity to write my thesis at the Communication Systems Group and Muriel Franco for the supervision during the entire process. Thank you for the advice and feedback in each and every meeting in which we discussed my work. Your help was very valuable and avoided a lot of stress and unnecessary worries. Also, thank you for providing me such an interesting topic, which made it a lot easier and fun to work on. Furthermore, I want to thank my second supervisor Eder Scheid, who provided a lot of helpful feedback and helped me with the presentations and final delivery.

Contents

Chapter 1

Introduction

The online advertising market has been growing significantly in the past few years. Alone in 2019 the digital ad spending market is predicted to grow 17.2% to over $330 billion [1]. Alphabet and Facebook two of the largest corporations measured by the market cap, make almost all of their revenue through their online advertising services. But also many online web services providers found the online advertising to be very lucrative [2]. Some are even only able to operate their services through the generated revenue from online advertising on their websites. Ads do not only generate revenue for the operator but also for content creators on these websites. On Youtube, for example, content creator get paid based on the amount of ad views on their videos.
End-users however do not profit directly from advertising. They are often left with unwanted ads, pop-ups or even phishing. This internet advertising business model relies on a implicit agreement between the providers and the user online, even though the loading of advertising is solely creating disadvantages for the users. It does not only reduce the overall user experience but also increase loading speed and bandwith usage. Especially for limited bandwith usage plans, which are common for mobile users or in developing countries, the increased data usage can have potentially significant economical effects.

1.1 Motivation

In order to avoid unwanted ads different adblocking tools (e.g Adblock Plus, Pi-Hole) have been gaining in popularity to reduce negative effects experienced by the end-users [3]. The operators, however, are trying to find new ways to bypass those adblockers through product placements, cooperation with advertisers or even blocking adblock users from accessing their content.
We argue that the current business model on ads being a multi billion dollar market has different advantages and drawbacks for the different stakeholders. Thus, an in depth analysis of the drawbacks and advantages of the current business model is required.
In this thesis we are going to focus on several Adblocking tools like Pi Hole, Adblock Plus and Perfect Privacy and analyze the impact of such tools from a user-experience as well as from an economical point of view.

1.2 Description of Work

This thesis measures bandwidth waste to load unwanted advertising and conducts an in depth analysis in the negative effects on different type of users behaviors (e.g., mobile and domestic users) through loading of unwanted ads. To get the the necessary data a internet surfing simulator will be developed.
An internet user behavior simulator tool was developed to collect data from real Internet advertising and to analyze the performance of different adblock solutions. Also, we conduct a literature review on adblockers, thus, identifying the different types of adblockers available and their technical characteristics. Based on such review and the tool developed, an in-depth data analysis was conducted and a discussion provided regarding the benefits and drawbacks of such tools for both end-users.
Finally, all the findings are discussed and documented written, detailed and visualized manner.

1.3 Thesis Outline

The thesis is structured as follows. Chapter 2 provides the background of ad blocking technologies, important tools available, and related work. In Chapter 3 the methodologies needed, the requirement analysis, the design and implementation of the internet surfing simulator tool is being documented. In Chapter four we analyze the data collected and providing a discussion regarding technical and economic aspects. Chapter 5 concludes this thesis and provides directions on future works.

Chapter 2

Background and Related Work

This chapter introduces the theoretical background and related work. Background of advertising and its use on the internet is given in Section 2.1. Next, a selection of different adblocking tools as well as a the techniques they use to block advertisement are presented. Finally, in section 2.3, different adblocking techniques are discussed, and an overview of tools to block advertisement is provided.

2.1 Advertising

The history of advertising, began in the 17th century [4]. Although, printing was invented as early as the 15th century, mass duplication of advertising pamphlets did not occur. There was no possibility for a large distribution of advertising messages. But with the emergence of the first daily newspaper around 1650, the advertising had now opened up a big gate to be mass distributed [4]. In 1850, with the industrial revolution, the history of advertising went into another phase. Advertising was so far mostly based on factual aspects, but from 1870 on, advertising started to become offensive and was targeted to certain groups of people and aimed at social classes. In 1900 there was a big boom in advertising, as companies realised that good advertising can lead to significant increases in sales. As a result advertisement texts and posters became larger and more noticeable [5].

The history of advertising continued in the 20th century. Advertising was present in many new technologies invented in this era. Whether it was television, radio or movie theaters, advertising always played an essential part in it and enabled these industries to exist with the revenue it generated.

Today, advertising is one of the driving forces of our consumer and society growth. Companies such as Apple, Adidas, Nike or RedBull spend billions of dollar in advertising and have created long-term relationships with their customers through the use of smart marketing. The internet offers companies an ideal platform to advertise their products tailored to their target group by collecting data based on their surfing behaviour. Whether on TV, on the Internet, via digital signage or in newspapers and magazines - advertising is part of it. It has been informing and entertaining for the past centuries and is playing an essential part in the century of digitization.

2.1.1 Advertising on the Internet

There are two significant advantages for companies to advertise on the internet: *(i)* displaying advertising online is usually cheaper than traditional channels and *(ii)* the success of advertising campaign can be precisely measured by using analytic tools such as Google Analytics to measure the advertising ROI. For this reasons and the ease to buy advertising spaces on the internet, it has gained in huge popularity over the past few years. In 2019, the digital ad spending market is predicted to grow 17.2% to over $330 billion USD [1]. Thus, the advertisements are spread in the Internet in different ways and platforms, such as news websites, social media, web applications, and content-providers.

There are several types of advertisements that are commonly used on the internet [7]. Currently, different mechanisms are available to customize advertisement according to the target group, which means that the type and content of advertising can vary according to the profile of the user accessing the Internet, thus resulting in a more effective and lucrative advertisement strategy.

Although the advertisement can be created on-demand according to the market needs (e.g., product placement and sponsorships), the most common advertising types used on the Internet are [7] (*cf.* Figure 2.1):

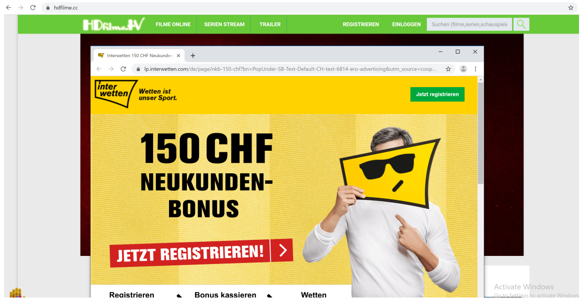
(a) **Banner ads**: The advertising is integrated as a graphic or animated file, usually in a GIF or Flash format, into the website. These Banners link to the advertisers website as hyperlinks. Banners can be embedded directly into the page, but also sometime lie over the page. They are the most common way of advertising on the internet but have been losing in importance to video advertising [6].

(b) **Pop-ups**: Pop-ups are used to display additional content or to request a specific interaction in an additional windows. Typically, pop-ups ”break” web-pages by covering other parts of the user interface. They are often seen as the most annoying type of advertisement on the internet, since the windows often cover up the whole page and have to be closed separately.

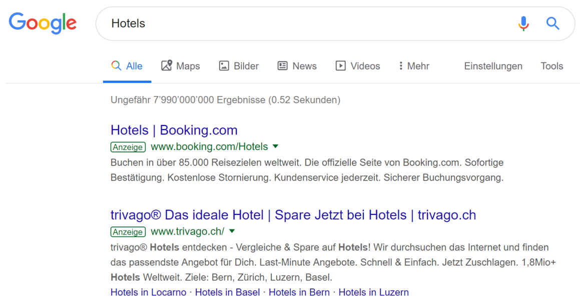
(c) **Text link ads**: In-text advertising turn individual words or phrases in the text into links. Usually, these links appear in a different color from the rest of the text. Google search is the most popular example of text link ads. The first few results shown in a google search are often text link ads, colored differently than the other results.

(d) **Video clips**: Like classic banner ads, commercial video clips are displayed on web pages or matching other editorial video content. They have been gaining large popularity with the increase of streaming platforms and social media. The most popular way for video advertisements are short ,5 to 15 seconds, sometimes skipable videos, played before video content is displayed.

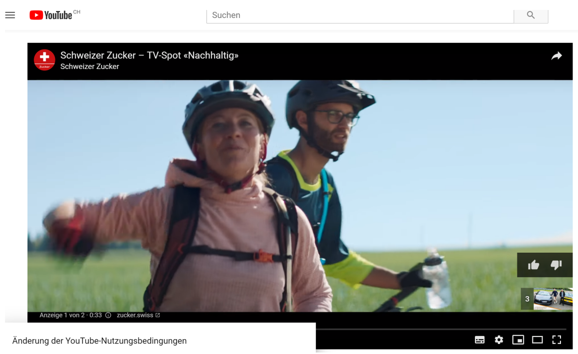
(a) Banner Ads



(b) Pop-Up



(c) Text-link ads



(d) Video Advertisement

Figure 2.1: Examples of Advertisements Types on the Internet

2.2 Adblockers

The presence of ads can impact the user experience negatively while surfing on the Internet. This has an impact on not only the loading speed of web-pages, but also on the internet bill of the user in certain cases (e.g., limited mobile data plan). Researches suggest, for example, that people avoid advertising messages on the Internet because of perceived ad clutter [7]. Although some consumers may continue to click on the ads they find useful, many consumers do choose to avoid that, because of an aversion to the high amount of advertisements on the Internet. Currently, although the ad cluttering on the Internet is reducing the collective effectiveness of Internet advertising, the content providers are still overloading pages with advertisements. Thus, in absence of a mutually-agreed procedure for opting out of advertisements, many users resort to adblocking tools. Studies have shown that almost 25% of the internet users have some sort of adblocking tool installed [3]. This trend is continuing and the overall distribution of adblockers is growing faster than ever, due to the increase of mobile adblockers in the past few years [10].

2.3 Adblocking Tools

Different tools are available to deal with and block advertisements on the Internet. In summary, three main groups of adblockers can be identified:

- **Virtual Private Network (VPN) with dedicated adblockers**: Some VPN services come with a built-in functionality that protects the user from advertisements.
- **Browser Extensions**: Third-party browser extensions that block advertisements directly in the browser. There are both free and paid applications that work with most web browsers.
- **Network-wide**: A dedicated adblocking router or a virtual router, for example, can be set up with a DNS filter that will automatically block requests from ad domains that are known to provide adware or tracking services.

It is worth to say that there are also some other types such as browser with built-in adblockers, but those are used by a minority of users and are in essence preinstalled adblocker extensions.

2.3.1 Pi-Hole

Pi-Hole is a network wide adblocking tool, which sets up a Domain Name System (DNS) server and handles all DNS requests generated from your home network [8]. Pi-Hole will deny all requests from adservers and thereby prevent the loading of advertisements. Pi-Hole does not modify the website or application's request to download any third-party scripts.
The advantage over other ad blocking alternatives is, that Pi-Hole blocks ads on network-level, which also allows for adblocking on non traditional devices such as Smartphones or TVs.

2.3.2 Adblock Plus

Adblock Plus is a browser based extension available for many of the most popular browsers. It was one of the first ad blocking extensions being available and is today one of the most popular extension for Chrome and Firefox with over 10 million downloads [9].

Adblock Plus has created big controversial claw back when it introduced Acceptable Ads in 2011, a functionality activated by default [13]. The initiative is designed to display less intrusive ads to visitors with activated adblockers on. For small to medium-sized web publisher it costs nothing to participate in the program. Large publishers, however, are required to pay to run Acceptable Ads on their sites. Therefore many large publishers have felt exorted and accused Adblock Plus of abusing their power to force them to pay for advertising.

2.3.3 Perfect Privacy

Perfect Privacy is a Swiss based VPN solution founded in 2008. They have been committed to the privacy and anonymity of its customers and have been one of the world's most secure

VPN providers [15]. Perfect Privacy provides its customers the TrackStop functionality, which is Perfect Privacy's own adblocker. TrackStop is not only able block annoying ads but also able to protect the computer from malware that can be injected via ad-servers, so-called malvertising [16].

2.4 Adblocking Techniques

Adblocking tools are generally speaking able to block most ads on the internet. This includes banner ads, pop-up ads, paid text-links and video advertisements. Additionally most tools provide some sort of privacy malware phishing protection for the user. The techniques used to block advertisements, however, can vary depending on the type of adblocking tool used [7].

Adblock Plus and Pi-Hole work using filter lists. These lists are publicly available and crowd-sourced. Users are able to add additional filter list or block single ad-domains. They prevent advertisement from actually being loaded by denying the request made from these ad-servers. VPN Blockers do also work with filter lists. Differently to Adblock Plus or Pi-Hole, it is not possible for the end-user to add additional filter lists, since the configuration is made by the VPN operator.

Most ads are loaded into web-pages via JavaScript. A few lines of JavaScript are placed on the site to query an ad-server which injects the site with the contextual advertisements. Adblock Plus attempts to prevent the JavaScript file from ever being loaded, whereas Pi-Hole and Perfect Privacy allow the JavaScript file to be loaded but block the request the script sends. Additionally if Adblock Plus fails to block the JavaScript, or if the ad is embedded through a different manner, such as hard coded onto the site, they attempt to use Cascading Style Sheets (CSS) to hide the advertisement.

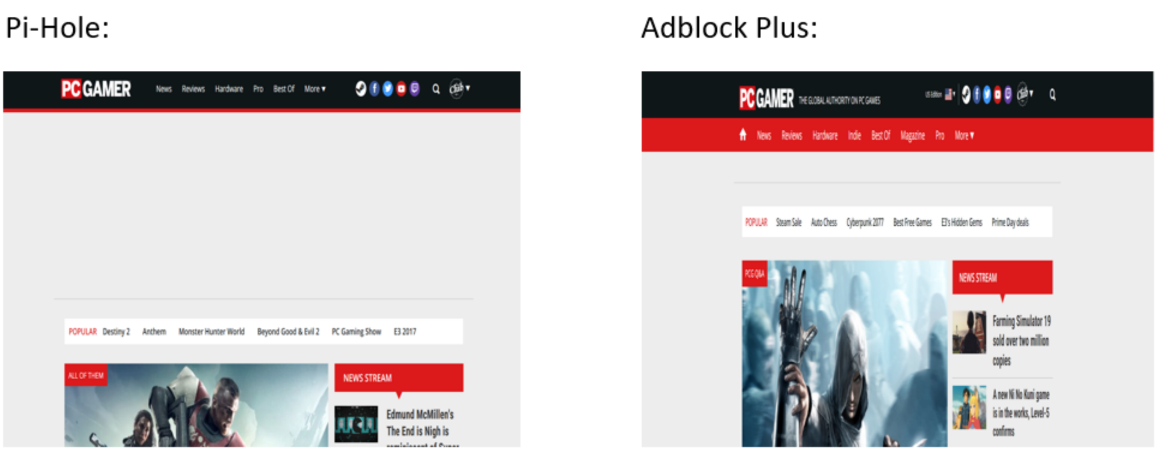


Figure 2.2: Difference between Pi-Hole and Adblock Plus

As seen in Figure 2.1, both adblockers show no advertisements on top of the article. However, a large empty grey area is visible when using Pi-Hole. This grey area is visible because of the way Pi-Hole works. Pi-Hole does only block the request from the ad-server on network level and does not use CSS to modify web-pages layout. Adblock Plus, however, cuts out the grey area entirely with the use of CSS and makes therefore the user-experience significantly better. As a result of the different approaches taken and the different techniques used, the effectiveness of such tools can vary significantly [11].

2.5 Impact of Adblockers

Adblocking is now a widespread practice among web users. The use of adblockers, however, results in significant losses for operators and web publishers. Estimates indicate that ad blocking translates into losses for publishers, reaching about $41 billion USD for 2016 [12].
These substantial losses for publishers, as well as the ever increasing adoption of adblocking, has led many website operators, who are fearing for their funding basis, to react. Some of them try to bypass the adblocker tools technically or ask the user, but do not force their visitors, to disable adblockers in order to be able to operate their sites.
A third route is taken by website operators by trying to completely exclude users, who are actively using adblockers, from using their services. So called anti-adblockers scripts are used on these websites, which detect any usage of adblockers and exclude adblock users from accessing the content. These users are than forced to deactivate their adblockers or otherwise will not be able to consume any content. Another tactic used often by the newspaper industry, who is strongly being affected by the switch from traditional newspaper to online content consumption and the usage of adblockers [14], is to only allow non adblocking user or premium users to access the content.

2.5.1 Anti-Adblock Blockers

Anti-adblock blockers are scripts used on websites, which detect whether a user accessing the site has an adbocker enabled. They check, whether advertisements on the page have been loaded correctly to detect the presence of adblockers and then block the content, which was wished to be accessed by the user in case the advertisements were not loaded [29].

Due to the increase of anti-adblock blockers on websites and users not accepting to being forced to consume unwanted ads, an opposite trend has been growing in the past few years. So-called anti-adblock killers have been developed and started to gain in popularity. These tools, often available as extensions, try to bypass anti-adblock blockers by trying to hide the active use of adblock. To counter this retaliation, anti-adblockers in turn try to detect and filter the scripts of these extensions. Thus, this back and forth has resulted in an escalating arms race between anti adblockers, adblockers and also anti-adblock blockers [29].

Chapter 3

User Surfing Behavior Simulator

To measure the bandwidth waste of loading unwanted advertising on the internet and to conduct an analysis of the impacts of these advertisements on users, a tool that simulates different internet surfing behaviors to generate traffic and data in different scenarios had to be implemented. In this chapter we are going to have a look at the requirements, design and architecture, the technical details of the source code and the final result of the implementation. Furthermore we are discussing the limitations and further possible applications and use-cases that can be based on the tool.

3.1 Requirements

Three initial main requirements that the tool needed to fulfill, in order to conduct the analysis of unwanted advertising traffic out of a economic and user experience perspective were specified. Those requirements are:

- A user behavior simulator should be able to generate real advertisement traffic to be blocked by adblockers.
- The simulator should be able to generate traffic for different user surfing behavior's (*e.g.,* mobile or desktop).
- The simulator should be able to be used with different adblocker tools such as Pi-Hole or adblocking extensions.

Further requirements such as automation of the tests were being added during the duration of the thesis with the aim to improve and expand the tool to additional use-cases. With the final implementation, all of the above mentioned requirements were able to be fulfilled.

3.2 Initial Steps and Design

The first step was to understand and research what for possible implementation methods existed to implement such a tool. First potential solution methods were data and statistics provided by the adblockers tools themselves. Pi-Hole, for example, provides statistics on their web interface and their API. These statistics, however, are in no interest for our tool as they do neither show performance statistics in regards to website loading speeds nor bandwidth usage data. Other possible methods were logs or cache analysis of adblockers and web-pages. After further research and tests it showed that this approach would have been too time-consuming and not effective. As these implementation methods seemed not to be achieving the goal, further research needed to be done.

After coming across different libraries and frameworks such as selenium, the following approach was defined: creating a simulator tool, that generates internet traffic based on a file containing a predefined list of URLs. These URLs should be automatically opened in a separate browser. While the browser navigates through the list of web-pages, the tool should simultaneously measure the amount of bandwidth used and time needed to open all web-pages. By running this simulator tool with different adblockers installed on the browser or adblockers configured on network level, it is possible to conduct an analysis of different adblockers. For this, in a first moment, a conceptual implementation was done in order to proof the feasibility of the work and provide initial results. After that, a refined implementation was provided taking into account initial evaluations and feedbacks.

The conceptual implementation was done with the help of the selenium web driver framework. Selenium is for automating web applications for testing purposes. With selenium it was able to navigate in chrome within the Python script and generate traffic. The only other essential part needed was a way to measure the bandwidth usage. After further testing and research, the psutil library was found to be providing the functionality needed. Those libraries and frameworks were used to implement a tool that fulfilled almost all requirements. However, one challenge was using the different ad blocking extensions with the selenium web driver framework. The reason for this issue was due to the way selenium creates a connection with a Google Chrome instance. Selenium does open a separate entirely new instance of Chrome every time the tool is being started, rather than using the normal installed Chrome browser on the computer to establish a connection. This means that additional extensions for the Chrome browser need to be installed within the Python script and any configuration changes for these extensions such as changing the user agent for simulating mobile user behavior or browser configuration changes needed to be done within the Python script. This resulted in new issues as it was rather hard to almost impossible to configure some of the extensions in the Python script.

3.3 Implementation

The final implementation was similar to the conceptual implementation with selenium. However, a different library to connect with the Chrome browser to resolve the issue previously identified during the conceptual implementation when installing extensions and configuring them. The solution for this problem was to use the Chrome DevTools protocol, which supports all necessary functions such as navigation and receiving event notifications. With the help of the PyChromeDevTools library, the simulator can use the Chrome DevTools within a Python script. Based on that, all requirements were able to be fulfilled and resulted in further advantages.

The additional advantages of this implementation are:

- Simple installation and configuration. Only thing needed is to add a remote debugging port to the Chrome instance. After this the script tool can create the connection with Chrome.
- The Browser can be configured in different ways.
- Tool can be be easily extended with new functionalities and can be used for several other use-cases.

For the final implementation, Python (Version 3.7.3) was choosen as the programming language, Chrome as the web browser to generate traffic on and PyChromeDevTools to interact with Chrome within a Python script. Also, the Python libraries psutil and threading were used to calculate and measure the bandwidth and the performance of the adblocker tools. In addition, the libraries xlsxwriter and xlrd were used to export the results as a spreadsheet. More background details and project-decisions concerning all technologies selected for the implementation are provided below.

3.3.1 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It an easy-to-learn yet powerful programming language with efficient abstract data structures and a simple effective approach to object-oriented programming. With its elegant syntax and dynamic typing, it is well-suited as an interpreted language for both scripts and rapid application development [17]. It was chosen as the programming language for the simulator tool due to the availability of all necessary frameworks and libraries.

3.3.2 Chrome

Google Chrome browser is an open source web-browser for accessing the World Wide Web and running Web-based applications.
The Google Chrome Web browser is based on the open source Chromium project. It is available for Linux, Windows, Mac Os X, Android and iOS operating systems [18]. Thanks to it's reliability, simple to use navigation and performance, it is now the world's most popular browser [19].
Chrome was chosen as the web-browser for the tool as all the adblock extensions are supported and all necessary API's exist to integrate it within a Python script.

3.3.3 PyChromeDevTools and Chrome DevTools Protocol

The Chrome DevTools Protocol allows for tools to inspect, instrument, profile and debug Chromium, Chrome and other blink based browsers. Many existing projects currently use the protocol [20]. The Chrome DevTools protol is used to navigate in Chrome and generate internet traffic.
With the help of the PyChromeDevTools library, a Python module that allows one to interact with Google Chrome, it is possible to use the Chrome DevTools Protocol within a Python script.

3.3.4 Psutil

The psutil, process and system utilities, is a cross-platform library for retrieving information on system utilization and running processes on the computer (e.g CPU, memory, disks, network, sensors) in Python. It is useful mainly for system monitoring, profiling and limiting process resources and management of running processes. It implements many functionalities offered by UNIX command line tools and provides the necessary function to measure the computers bandwidth usage [21].

3.3.5 Tkinter

Tkinter is a part of the Python standard library which provides an Object Oriented interface onto Tk/TCL. This enables to build a cross platform GUI for Python applications without needing further dependencies [22]. It is a simple to learn Python framework and allows to create an user interface, to simplify and improve the user experience of the simulator tool.

3.3.6 Chrome User Agent

The Chrome User Agent is a Google Chrome extension that allows to quickly switch between user agents and test how different websites are displayed in different browsers and on different devices such as Mobile IOS, desktop Windows or desktop Mac OS.
The extension is used for the analysis of the effects of loading unwanted advertisements for mobile users by opening the mobile version of websites on a desktop PC.

3.4 Architecture and Design

The tool was designed in such a way that it is as easy to use as possible, fulfill all necessary requirement to conduct an analysis of adblockers and that it can easily be extended to other use-cases and scenarios.

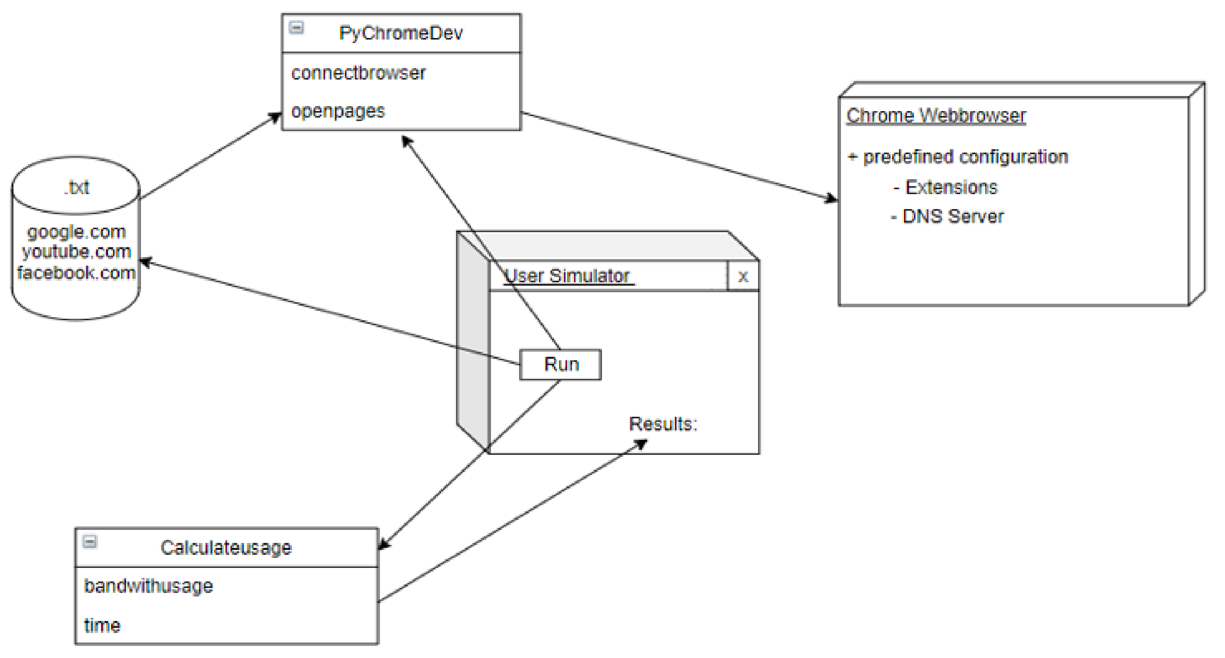


Figure 3.1: Tool Architecture

The tool consists of five main components (*cf.* Figure 3.1). A text file containing a list of URLs, an in advance configured Chrome Browser, the simulator tool User Interface (UI), and two background processes. First precondition for the tool is to have a Google Chrome browser opened. Second precondition for the tool is to have a simulator.txt file with a list of URLs. As the tool is started, a connection with the Chrome browser instance is established and the .txt file is being read in by the tool. As soon as the user starts the script with the button "Run Script" on the UI, the web-pages are started to be opened in Chrome and simultaneously measurement of bandwidth usage and time performance is done. At the end the results are being displayed on the UI and saved in an excel file.

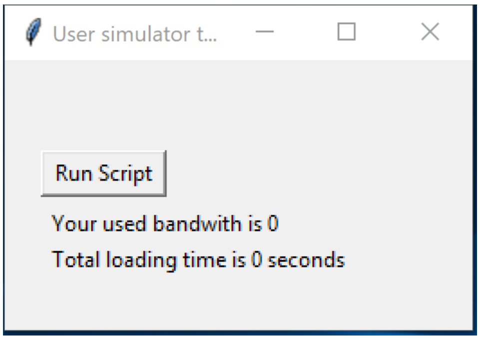


Figure 3.2: Simulator Tool

Figure 2.1 shows the Interface of the Simulator tool. The Interface has been kept as simple as possible, while showing as much relevant information as needed. It contains of one button to run the script and a section where the results of the tests are shown. While generating traffic a progress bar shows the current status on how many of the total amount of web-pages to be tested have been opened.

3.5 Implementation Details

This section gives an in-depth look on the source-code of the tool. Five main processes in Python, connection with the browser, initialization, generating traffic and bandwidth calculation are introduced. The presented source code snippets do not represent the whole source code of the tool.

3.5.1 Connection with the Browser

The tool uses the Chrome DevTools Protocol to instrument the Google Chrome web-browser. To be able to interact with the Chrome DevTools Protocol within a Python script, it is necessary to create a connection with a running Chrome instance. This is done with the help of the PyChromeDevTools library, as shown in Listing 3.1. First a new ChromeInterface object is created and assigned to the variable chrome. With this a connection has been established with Chrome and all the Chrome DevTools Protocols can be used on the chrome object in Python. After a connection has been established with the browser, the protocol Network.enable() and Page.enable() are called to enable network tracking and network events delivery to the client in order to be able to receive event notifications. Activating the notification events is necessary for receiving the notification of pages being fully loaded in Chrome.

```
import PyChromeDevTools

def createconnection(self):
    self.chrome = PyChromeDevTools.ChromeInterface()
    self.chrome.Network.enable()
```

```
    self.chrome.Page.enable()
```

Listing 3.1: Connection with the browser

3.5.2 Browser Configuration

A precondition for using the simulator tool properly is to configure the browser in advance. For the analysis of the effects of unwanted ads on bandwidth and performance a deletion of all browser data including cache is necessary to be performed before testing. In addition, it is necessary to deactivate the cache completely for automated testing. This is done by opening the Chrome DevTools and pressing the F12 button.

Furthermore, in order to generate traffic with the different adblockers, we need to firstly install the adblockers and activate them in Chrome. Also further configuration in the adblockers need to be done in advance in the browser (e.g adding filterlist, deactivating Acceptable Ads). In case of Pi-Hole the configurations are not done in the browser, since Pi-Hole needs to be configured on network level. For testing mobile user behavior the Chrome User Agent extension needs to be installed and configured.

3.5.3 Initialization and User Interface

During the initialization of the simulator tool the interface is being created with the tkinter library. The interface contains a button called "Run Script" and two widgets showing the results of each test. In a next step the connection with the browser is established by calling the function createconnection(), the .txt file is being read in with the function openfile() and the progressbar is created with the function createpro() (*cf.* Listing 3.2). Furthermore the excel sheet is created with the function xlsxwriter.Workbook('Results.xls').

```
import tkginter

class View(Tk):
    def __init__(self):
        Tk.__init__(self)
        self.title("User simulator tool")
        self.geometry('260x150+400+100')
        self.bRechne = Button(master=self,
        text="Run Script", command=self.run)
        self.bRechne.place(x=20, y=50, width=70)
        # Label
        self.lC = Label(master=self,
        text= " Your used bandwith is " + str(0)  + " mb")
        self.lC.place(x=20, y=80)
        self.time = Label(master=self,
        text=" Total loading time is " + str(0) +" seconds")
        self.time.place(x=20, y=100)
```

```
        self.createconnection()
        self.openfile()
        self.createpro()
        self.testnr = 0
        self.workbook = xlsxwriter.Workbook('Results.xls')
        self.worksheet = self.workbook.add_worksheet()
        self.workbook.close()

    def createpro(self):
        self.p = Progressbar(self, orient=HORIZONTAL,
        length=200, mode="determinate",
        takefocus=True, maximum=self.num_lines)
        self.p.pack()
```

Listing 3.2: Initialization and User Interface

3.5.4 Generating Traffic

First of all a predefined text file called simulator.txt has to be created and placed inside the projects folder. The simulator.txt files contains a list of URLs of the web-pages, that are to be opened by the tool. Every URL inside the text file is separated by a new line. After the tool has been initialized and the user has started the script, the tools iterates through every line of the simulator.txt file and opens the web-page with the function chrome.Page.navigate(url). This will instruct the Chrome instance to navigate to the specified web-page. The next necessary step before loading the next page is to wait until the previous page has been fully loaded in Chrome. This is implemented by waiting for the loadEventFired event, which gets triggered in Chrome when pages are fully loaded. This makes sure that the next web-page is opened only after the current webpage has been fully loaded. This includes all data and scripts running on the web-page. In case a website is being stuck on loading, a timeout after 60 seconds has been implemented.

At the same time with the start of opening the web-pages a timer is being started. The timer is being stopped after all the pages have been opened and the time is then returned.

In the end after the tool has iterated through the whole simulator.txt file, the timer is being stopped and the bandwidth calculation thread is closed. This is done by changing the boolean do_run inside the download_thread object. Afterwards the results are saved in the excel file by calling the function writetoexcel(). Listing 3.3 shows the function openfile() to read in the text file and the function berechne() to generate traffic with Chrome.

```
def openfile(self):
    self.f = open("simulator.txt", "r")
    self.num_lines = sum(1 for line in self.f)
    self.f = open("simulator.txt", "r")
    print(self.num_lines)
    print("opened file")
```

```
def berechne(self):
    print("start opening pages")
    self.start_time = time.time()
    self.openfile()
    for x in self.f:
        print("for page in file")
        self.chrome.Page.navigate(url = x)
        self.chrome.wait_event("Page.loadEventFired",timeout=60)
        self.p.step()
        self.update()
    self.elapsed_time = self.format(time.time()-self.start_time)
    self.time = Label(master=self, text="Total loading time is "
    + str(self.elapsed_time) + " seconds")
    self.time.place(x=20, y=100)
    self.download_thread.do_run = False
    print(self.elapsed_time)
    self.writetoexcel2()
```

Listing 3.3: Generating Traffic

3.5.5 Bandwidth Calculation

The psutil library allows to retrieve information on running processes and system utilization. To measure the total amount of bandwidth used, while the tool is generating traffic, a thread needs to be created. This is done with the function threading.Thread(target=self.calculateusage) as seen in Listing 3.4.

```
import threading

def run(self):
    self.reset()
    self.download_thread = \
    threading.Thread(target=self.calculateusage)
    self.download_thread.do_run = True
    self.download_thread.start()
    self.berechne1()
```

Listing 3.4: Threading

The functions of interest from psutil are net_io_counter().bytes_sent and psutil.net_io_counters().bytes_recv. net_io_counter().bytes_sent returns the total amount data sent by the computer and psutil.net_io_counters().bytes_recv returns the total amount of data received by the computer. Both these results of the functions are calculated since the start of the computer and are returned in kilobytes.

To start calculating the total amount of bandwidth used during the time frame the tool generates traffic, the initial amount of data packages sent and received by the computer, before the tool is generating traffic, needs to be subtracted. Thereafter, while traffic is generated, the total amount of bandwidth usage is calculated in the while loop as shown in Listing 3.5. In the end the results are converted to megabytes with the function convert_to_gbit() and returned.

```
import psutil

def calculateusage(self):
        print('yes1')
        self.old_value = 0
        self.once = True
        while self.run:
            self.new_value = psutil.net_io_counters().bytes_sent
            + psutil.net_io_counters().bytes_recv

            if once:
                self.once = False
                self.old_value = self.new_value
                send_stat(self.old_value)
                time.sleep(1)
            else:
                send_stat(new_value - old_value)
                time.sleep(1)

    def convert_to_gbit(self,value):
        return (value / 1024. / 1024.)

    def send_stat(self,value):
        print("%0.3f" % convert_to_gbit(value))
```

Listing 3.5: Bandwidth Calculation

.

3.5.6 Limitations and Future Use Cases

Due to the architecture and design of the tool, also some certain limitations exist. The total bandwidth used during the run of the script is calculated by the total amount of data sent and received by the computer. This means that the total amount of bandwidth used calculated by the tool is not fully representing the actual traffic generated by the web-pages, as some background processes on windows for example always use very little amounts of data. During tests, however, it showed that the background data usage is so low that it is negligible for the bandwidth usage analysis. There were some very rare cases background downloading of other tools occurred, which distorted the results significantly and made them unusable. That is why it is important to make sure that during the time the tests are performed all unneeded applications are closed, no application is using performing background downloading and no other actions are performed on the computer, as otherwise the results do get distorted. For this reason it is recommendable to run at least two tests on the same set of URLs.

Another limitation is the simulation of the mobile user behaviour. The simulation is performed by changing the user agent of the desktop chrome browser to a mobile browser user agent. This tricks websites to loading the mobile pages of a website. But however, these results can be inaccurate, as still more data is being sometimes loaded, due to the bigger screen and different layout. Additionally it is not possible to simulate traffic generated by Mobile Applications, which represent a substantial amount of data used on mobile.

Thanks to its architecture and design, the tool can not only be used to measure the bandwidth usage and performance of adblockers, but can also be used for other use-cases and scenarios. Speed tests between different computers can be performed, loading page of single sites can be calculated, performance impacts of other extensions can be measured and much more. Additionally, by extending the code, new measurement variables can be added to be measured by the simulator tool.

Chapter 4

Evaluation and Analysis

Previous studies have shown that adblockers such as Adblock Plus do indeed significantly reduce the number of HTTP and HTTPS requests when browsing the internet [31]. The question that arises is how the reduced amount of requests translate into real improvements of the user-experience. Therefore, we analyze and evaluate different adblocking tools in regards to performance and bandwidth usage with the help of the simulator tool. Also, we investigate the impact of non-intrusive advertisements and test whether websites block their content when when accessing them with an adblocker. Additionally, the potential cost saving a user with a limited data plan could have by reducing the bandwidth usage with an adblocker are presented.

4.1 Setup

Using the simulator tool with the same list of URLs used in this thesis on a different computer with other specifications can result in other results in regards to web-page loading speed, as page loading speed is strongly dependent on CPU speed. Bandwidth usage and the relative performance differences between the adblocker tools, however, should be similar to the results obtained in this thesis. Also the internet downloading speed and uploading speed could potentially have an influence on the results. The tests were performed on a desktop Personal Computer (PC) with following specifications:

Table 4.1: Desktop PC Specification

Processor	Intel Core i5 6600K
RAM	16 GB
Graphic Card	AMD Radeon R270x
Operating System	Microsoft Windows 10 Pro
Google Chrome	Version 75.0.3770.142
Internet Speed	100 MB for both down/up

Since the goal was to perform general easily reproducible tests, no additional adjustments such as adding additional filter lists to the standard installed filter lists were made. The configuration for each one of the adblocker tool used during the evaluation are:

- **Vanilla**: We use the term Vanilla to describe tests being performed with no adblocking tool in use.
- **Pi-Hole**: The standard domain blocking lists from Pi-Hole were used. Contains of StevenBlock, MalwareDom, Camelon, Zeustracker, DisconTrack, DisconAd and Hostsfile.
- **Adblock Plus**: The standard filter lists were used. The by default activated Acceptable Ads function was deactivated. In an additional test the Acceptable Ads function was activated. This is mentioned separately.
- **uBlock Origin**: The standard filter lists were used. No additional changes made.
- **Perfect Privacy**: Perfect Privacy was used with the TrackStop feature enabled. Basel was chosen as the VPN location, since it was the nearest location. Between the two available VPN protocols, OpenVPN and IPSEC, IPSEC was chosen.

4.1.1 SimilarWeb

SimilarWeb Ltd is an IT company founded in March 2009. Based in London, SimilarWeb offers international web analytics, data mining and business intelligence services. It leverages big data technologies to measure, collect, analyze, and deliver user engagement statistics for both web-pages and mobile apps [30].

SimilarWeb provides the data on the most visited websites, which is used to simulate the different user behaviours. Not only global visit metrics are available, but also specific visit metrics of countries around the world including Switzerland. Furthermore, SimilarWeb provides visit and engagement metrics for websites visited on desktop as well as mobile separately.

4.2 Desktop User

In this section we evaluate and analyze different adblocking solutions available for domestic users on desktop PCs. The performance differences as well as the impact on bandwidth usage of those solutions are analyzed with the help of the simulator tool. Furthermore, we investigate, what the impact of white-listing of non intrusive-advertisements in blacklists with such functions as Acceptable Ads is and how many websites bar users from accessing their content by detecting adblockers in use.

4.2.1 Methodology

The list of websites with which traffic is generated is based on the website visit metrics from SimilarWeb. The top 50 sites visited in Switzerland for the month of June were used, since they represent the majority of the internet traffic [23].

For each of those 50 sites an additional different web-page from the same website domain was added, which resulted in a total of 100 URLs. Similarweb does only provide the domains and not specific visited sites, so those additional sites were selected randomly, but as a rule, pages have been selected which are considered to be visited frequently on those sites. For example Google search results for Google.com, news articles for news websites etc. This was done so that the list of URLs did not only consist out of homepages and actual user surfing behavior is better represented.

The most visited sites of Switzerland were used rather than the global most visited, since countries with a higher population have a stronger influence in the global visit metrics. Additionally, international corporate websites are over represented due to their awareness and global presence.

The simulation with the tool was performed on Chrome with each adblocking solutions with the same list of URLs three times. Out of the three results the average was calculated and then used for the analysis. This approach was chosen to ensure that the results were not distorted through background activities. In addition, fluctuations in the results could be absorbed better, since it showed in previous tests that the results can differ by a few megabytes and seconds in each run.

4.2.2 Performance

The total average amount of time needed to load all 100 web-pages in Chrome with the Vanilla method was 187,1 seconds. Compared with the other methods, it was the worst performing one, as it can be seen in Figure 4.1.
Perfect Privacy with its adblocking functionality was able to lower the total loading speed by an average of 5 seconds, although data encryption and decryption used by VPNs are CPU intensive operations and slower the internet loading speed [27].
Adblock Plus and uBlock Origin were both also able to load the pages faster. There were however some noticeable differences between these two adblocker, even though they belong both to the same type of adblocking tool. Adblock Plus loaded the pages in 182,2 seconds, which is only about 6 seconds faster than the Vanilla method, whereas uBlock Origin loaded the same web-pages in an average of 157 seconds, 14% faster than Adblock Plus and 16% faster than Vanilla.
This means that adblocking extensions can improve the loading speed of websites by quite a large margin. The differences, however between the adblocking extensions can be significant. Therefore, it is recommendable to select, out of the dozens adblocking extension that exist, one which has proven to be efficient and fast. In this case it was uBlock Origin clearly outperforming AdBlock Plus.
The best result was able to be obtained with the use of Pi-hole. It took only 148,6 seconds

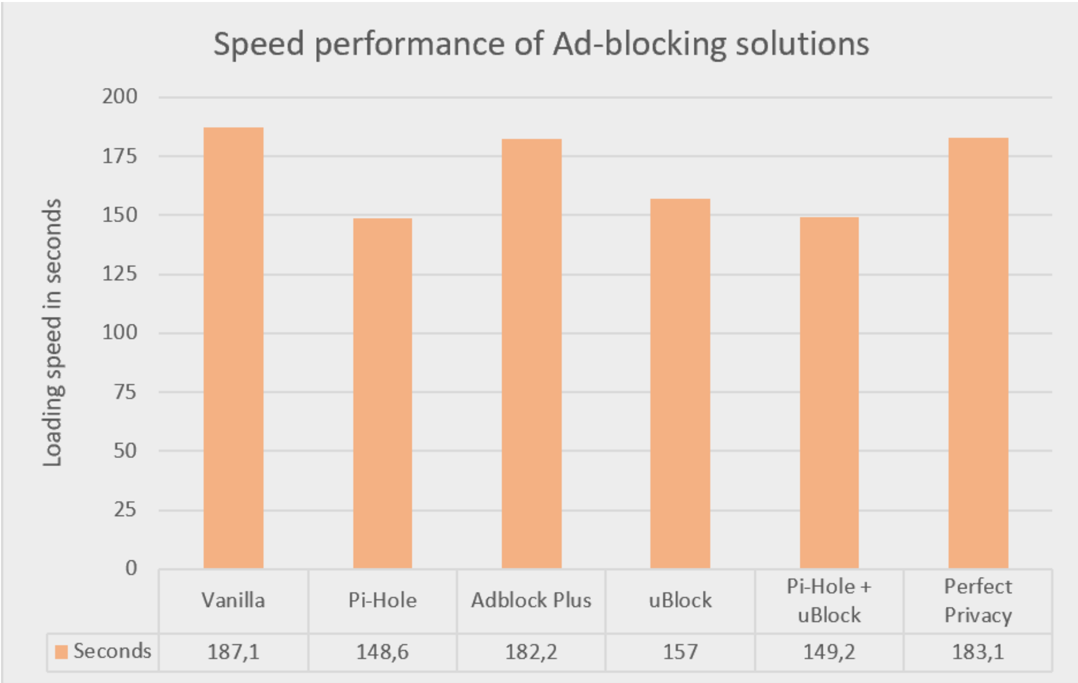


Figure 4.1: Total Loading Speed of Adblocking Solutions and Vanilla Browsing on Desktop

to load all web-pages, which is an impressive 21% faster than the Vanilla method. The fact that Pi-Hole is not executing any code in Chrome and is blocking the requests on network level could be reasons for the major performance improvements. In an additional test, Pi-Hole was used in combination with the better performing adblock solution, in this case uBlock Origin. Using Pi-Hole on a network level together with a adblocker extension, is a popular method for users to combine the advantages of each tool. This resulted in a total loading speed of 149.2 seconds, about half a second slower than using Pi-Hole as a single solution. The additional processes executed did not have any noticeable impact in this case.

We can conclude that the loading of advertisement does have a significant effect on the loading speed of web-pages. Depending on the type of adblocking solution used those improvements can vary significantly. Pi-Hole proved to be the best solution to improve web-page loading speed, Adblock Plus and Perfect Privacy showed to be worst, but still with all adblocking tools speed improvements were able to be noticed.

4.2.3 Bandwidth

Loading all 100 web-pages in the Google Chrome browser without any adblocker installed resulted in an average bandwidth usage of about 214.3 megabytes, as shown in Figure 4.2. For both Adblocker extensions, Adblock Plus and uBlock Origin, the total bandwidth usage was lower than with Vanilla. As in the previous performance analysis, differences between the adblock extensions was noticeable. Adblock used 189.9 megabytes, saving about 12% bandwidth, whereas uBlock Origin used 179,2 megabytes, saving about 17% bandwidth. Pi-Hole was the best performing solution used in this test, when looking

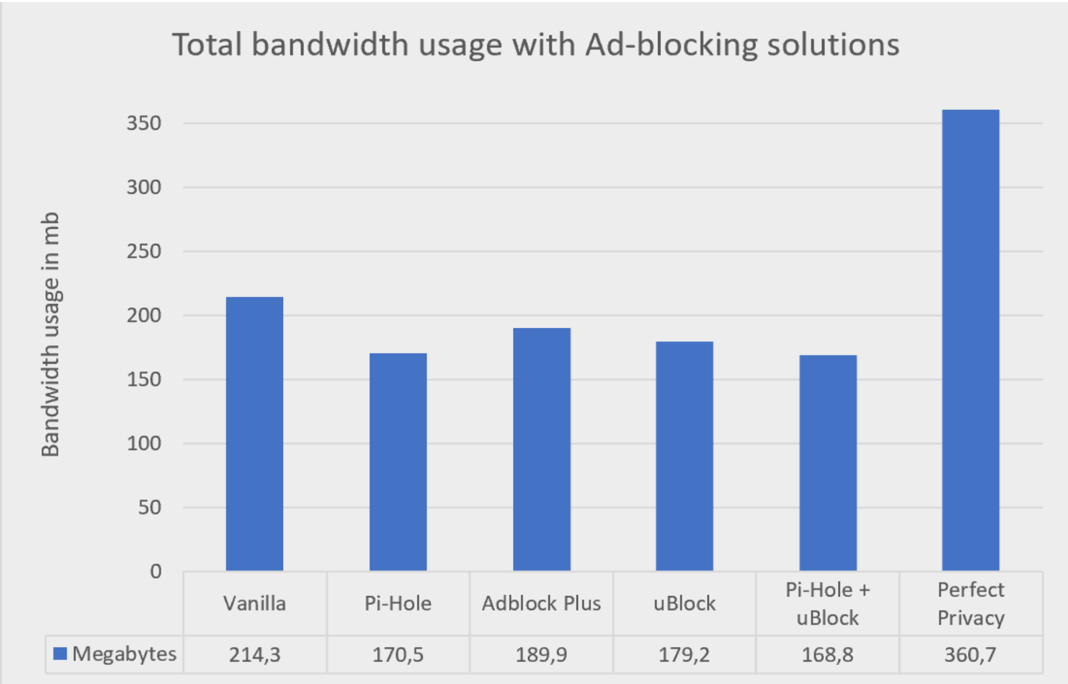


Figure 4.2: Total Bandwidth Usage of Adblocking Solutions and Vanilla Browsing on Desktop

at single solutions. With an average bandwidth usage of 170.5 megabytes, it was able to outperform every other single method. The saved bandwidth in comparison to the Vanilla method was 43.8 megabytes, a reduction of 21% percent. In combination with uBlock Origin, it was able to reduce it even to 168.8 megabytes.
Using the VPN solution, Perfect Privacy, increased the actual data usage by over 55% to 360.7 megabytes compared to Vanilla. This is by far the highest bandwidth usage out of all methods used in this test. The higher bandwidth usage is due to the encryption used to protect data transferred by VPNs. The encryption scrambles the data so that only the computer and the VPN server can read it. Once a file or piece of data is encrypted, it takes up more space and data than the unencrypted file would do. This is called encryption overhead [25]. In our case 256-bit IPSEC was used, which is amongst the protocols with the highest overhead. Other protocols such as PPTP 128-bit generate less overhead, but generally speaking a 5-20% increase can be expected with every VPN protocol [26].

It can be concluded that the single usage of Pi-Hole or in combination with an adblocking extension such as uBlock Origin is the best solution for users to minimize their bandwidth usage. In larger household with a slow internet speed connection and many unique users surfing the web at the same time, even improved loading speeds could potentially be noticed.
The use of a VPN with built-in adblock function is the worst ad blocking tool to minimize data consumption. VPNs, however, provide other benefits for which they are being mainly used, such as privacy through the encryption or remote employee access to company resources [34].

4.2.4 Acceptable Ads

Next, we look at the relevance of non-intrusive advertisements, often called Acceptable Ads. On one hand, adblockers threaten the financial backbone of the web and on the other hand, they also ensure some balance by preventing advertisements from becoming too intrusive [31]. A possible solution to solve this problem are non-intrusive ads list, which white-lists non-intrusive adverts that are usually blocked by blacklists. Adblock Plus has such a functionality enabled by default. But what is the effect of this list in regards to performance and bandwidth in our case? Given our vantage point and our ability to classify traffic the same way Adblock Plus does, we investigate the impact of the white-listing of these non-intrusive advertisements by performing a test on the same set of URLs used previously, but having the Acceptable Ads function enabled in Adblock Plus.

Table 4.2: Impact of Acceptable Ads

Metric	Vanilla	Adblock P. + Acceptable Ads	Adblock P.
Bandwidth	214,3 MB	208.7 MB	189,9 MB
Performance	187.1 s	189.7 s	182.2 s

Surprisingly, the total amount of bandwidth used and the total time needed to load all pages was significantly higher with Adblock Plus and the Acceptable Ads function enabled than having it disabled, as seen in Table 4.2. The results are comparable to those of the Vanilla method. Our results contradict the findings of E. Pujol et al., where it showed that the total amount of white-listed ad requests from the blacklist is only 15.3% [31]. We can assume that in our case acceptable ads have a much larger impact due to the fact that our set URLs does consist out of a much smaller set, in fact 100 pages from the top 50 websites, where a majority of sites are owned by larger companies, that can afford the administrative work and sometimes the additional financial cost related with participating in this program. In point of fact, according to the Financial Times [32], some of the largest internet corporations including Amazon, Microsoft and Google, pay money to Adblock Plus to be excluded from their blacklist, which would support this assumption.

4.2.5 Blocked Web-Pages

Next, we analyze how many of the sites used anti adblock scripts to exclude adblock users from accessing the content on their sites.
During our test, three out of the 50 sites blocked access to their content, when using Adblock Plus or uBlock Origin. Those were 20min.ch, tagesanzeiger.ch and blick.ch, news paper websites, an industry heavily dependent on revenue generated from advertisements [14]. Studies showed that 6.7% out of use top 5k Alexa sites use some sort of anti-adblock scripts [28]. In our test, it were 3 out of top 50 websites using anti-adblock scripts, which equals to 6% and is close to the results obtained in the study.

At a high level, anti-adblockers check, whether advertisements on the page have been loaded correctly to detect the presence of adblockers [29]. In this case, however, the content was not blocked when using Pi-Hole or Perfect Privacy as the adblocker tools, although the adblockers blocked ads on those respective pages. We can assume, that the anti-adblocker scripts used on those sites, detect only whether an adblocking extensions has been installed, JavaScript code is executed to block ads or CSS is used to modify the web-page. Thus on these sites, adblocker tools, that do no modification on browser level, are not detected.

There are however other sites such as forbes.com that even detect the usage of network level adblockers (*e.g.,* Pi-Hole and VPNs) and block the content for the user to access. This means, that a future rise of network level adblockers may force content providers to implement more advanced scripts that are able to detect network side adblockers.

We can conclude that a relative low amount of sites use anti-adblocker scripts to bar adblock users from accessing their sites content. These are industries that generate a majority of their revenue by selling advertisement spaces. Network adblockers can be advantageous when trying to bypass those anti adblocker scripts. However, they do not guarantee access to those sites, since there do already exist more advanced anti-adblocker scripts that are able to detect use of such adblocking tools.

4.2.6 Video Advertisements

One aspect which could not be covered by the simulator tool but is crucial when talking about advertisements and the impact on the user-experience are video ads and video traffic in general. Video traffic is a substantial amount of the data traffic generated on the web. In point in fact, almost 75% of the whole global internet traffic is video traffic [35]. Video advertisements are therefore a major disruptive factor for users. Not only do video advertisements basically block the user from the content for several seconds, but they do also consume a large amount of bandwidth. Therefore it is also important to analyze how the adblocking solutions perform in regards to blocking video advertisements.
Pi-Hole and Perfect Privacy were not able to block video advertisements on many of the most popular streaming websites or social media sites such as Youtube, Twitch or Facebook. These tools block the DNS queries and since these sites use the same domain to load the advertisements, the advertisements can not be blocked without blocking the website itself. This the nature of blocking by domain, which is a fairly monolithic solution which is great for some purposes and not great for other.
The adblocker extensions on the other hand are all able to block video advertisements. Especially for this reason, it is recommendable for users wishing to use Pi-Hole, to be using it in combination with an adblocker extension.

4.3 Mobile User

In this section we evaluate and analyze different adblocking solutions available for mobile users. First different adblocking solutions available for mobile users are evaluated and analyzed in regards to the impact on performance and bandwidth usage with the help of the simulator tool. Second, we investigate whether websites bar mobile users from accessing their content by detecting if adblockers are in use the same way they do on desktop. In the end, based on the results obtained in the bandwidth usage test, we show how data plan prices compare around the world and analyze how much money users with limited data plans could on average potentially save by using an adblock solution.

4.3.1 Difference Desktop and Mobile

Almost almost all of today's most popular web-pages are adjusted for mobile devices. These mobile optimized websites are an alternate version of a website, which are optimized for the smaller screen as well as the touch based navigation.
The first test was to measure how those optimizations generally translate into less bandwidth usage and performance improvements for mobile users. For this the same list of URLs was used as in the previous test for desktop PCs. The user agent, however, was changed to mobile, so that mobile version of the web-pages were being loaded. The results of the tests were as follow:

Table 4.3: Bandwidth and Performance Difference between Mobile and Desktop

Type	Desktop PC	Mobile Device
Bandwidth	214.32 MB	126.6 MB
Performance	190.1 s	148.27 s

The results, shown in Table 4.3, show that the bandwidth usage when loading the mobile pages of websites is significantly lower than loading the desktop version of the same webpage. Smartphone users benefit greatly from this, as the mobile devices generally have weaker processors, which results in slower website loading. Furthermore the users have often limited data caps when using cellular data, therefore the lower bandwidth usage is highly beneficial when browsing the internet with cellular data.
The question that evolves through this is if the relative amount of advertisements on those mobile pages stay the same or whether advertisements have a larger or smaller relative impact than they do on desktop. This will be investigated further in this section.

4.3.2 Methodology

Similarly to the previous test on desktop PC, a list of 100 web-pages consisting out of the top 50 most visited websites in Switzerland for the month of June was used. The difference here being however that rather than using the top 50 websites visited on desktop, the top 50 websites visited on mobile was used. One additional difference is that uBlock Origin was replaced, since it is not available on neither of the mobile operating systems. As a replacement, Adguard was chosen, one of the most popular iOS adblocker and Android adblocker extension.

It can be assumed that the results for the bandwidth analysis are accurate, since the amount of data being loaded should not differ whether the mobile web-pages are loaded on a smartphone or on a desktop PC, due to the fact, that the same content is being loaded. The performance evaluation however, should be taken with caution, since it is being performed by simulating mobile surfing on a desktop PC. It should still be able to give a direction on what to be expected when using adblockers on mobile devices. Moreover, since advertisements in mobile applications differ from browser ads and are not able to be tested with the simulator tool, we limit our analysis to web-pages only in this thesis.

4.3.3 Performance

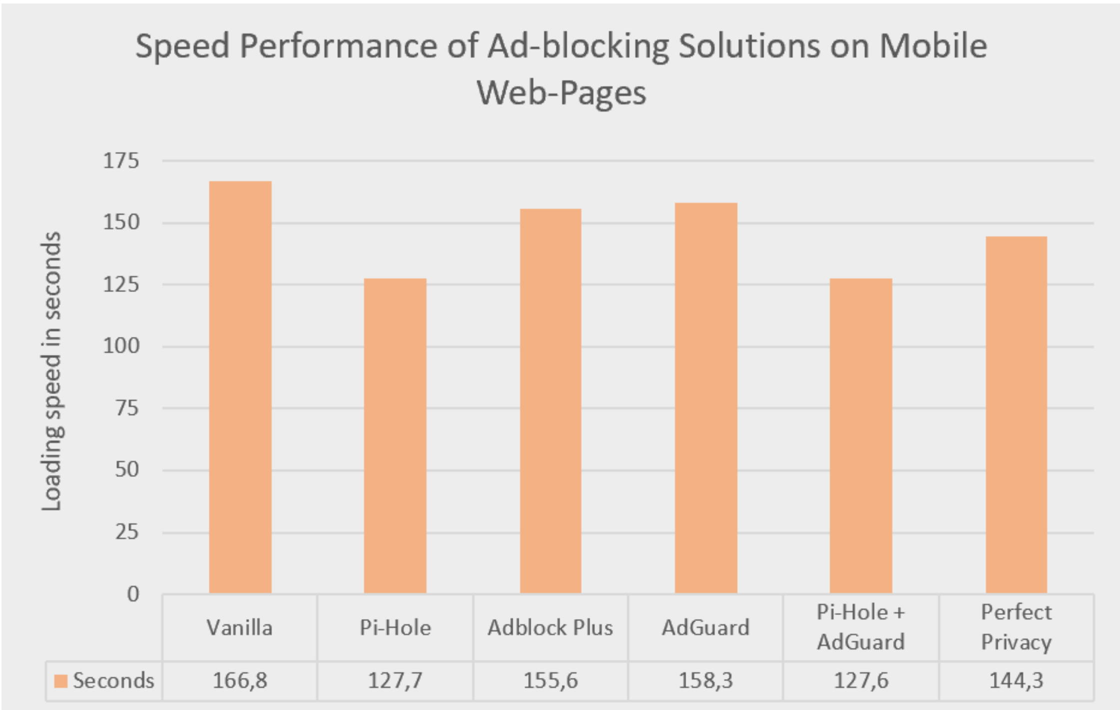


Figure 4.3: Speed Performance of Adblocking Solutions and Vanilla Browsing on Mobile

The performance result seen in Figure 4.3 show similar characteristics as the results obtained in the desktop test. Interestingly, Perfect Privacy was able to perform a bit better than both adblocking extensions with total loading speed of 144.3 seconds, a 14% improvement compared to the Vanilla method. This shows that a VPN solution with a dedicated adblocker can actually be a viable solution for mobile users trying to optimize their online browsing experience. Once again Pi-Hole was with a large edge the best performing solution. The relative improvements were with 24% for Pi-Hole just a little bit better than those obtained on desktop, where the improvements were about 21%. It is worth noting that Adblock Plus did a little better in this test with 7% faster loading to 3% on desktop, whereas Adguard did not match uBlock's performance improvements of 16% and was even slower than Adblock Plus, which previously showed to be much slower than uBlock Origin on desktop web-pages.

4.3.4 Bandwidth

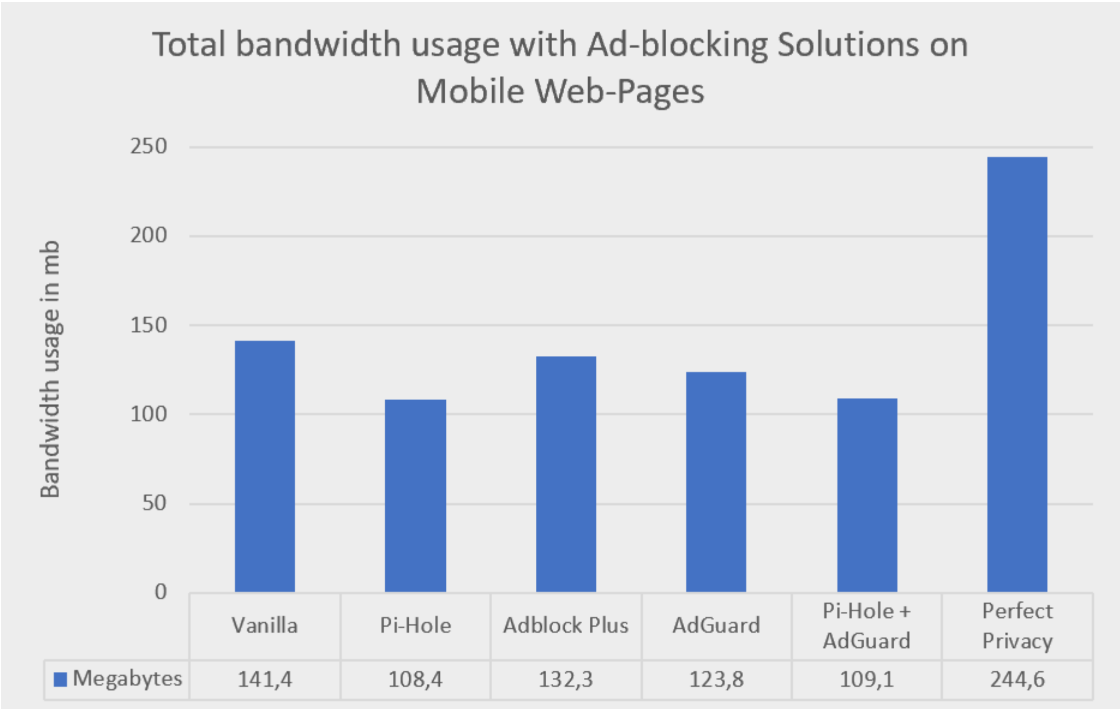


Figure 4.4: Total Bandwidth Usage of Adblocking Solutions and Vanilla Browsing on Mobile

Also for the bandwidth usage, the results were similar to the previous results obtained on desktop. Perfect Privacy consumed the most bandwidth in this test, for the same reasons as stated previously. In second place the Vanilla method was ranked with a total of 141.4 megabytes as shown in Figure 4.4.
The other three adblocking solutions were, as previously already obtained in the desktop test, able to lower the bandwidth usage by quite a margin, whereby Pi-Hole once again used the least amount of bandwidth with 108.4 megabyte, saving about 24% bandwidth. In contrary to the performance results where Adblock Plus performed performance wise a little bit better than Adguard, Adguard did better than Adbock Plus bandwidth wise with a total bandwidth usage of 123.8 megabytes to in comparison 132.3 megabytes with Adblock Plus.

All in all, it can be said that adblocker do save bandwidth noticeably. VPNs with dedicated adblockers here being however an exception, due to the way they work.
Major differences between mobile and desktop could not be observed by the amount of bandwith saved by adblockers. Thus one can say that the relative amount of advertisements on mobile and desktop are about the same.

4.3.5 Blocked Web-Pages

In contrast to the results obtained in test for domestic users on desktop PCs, none of the sites blocked access to the content, when using any of the adblocking tools. Even 20min.ch and blick.ch, included in both tests, did not block the content when accessing the mobile version of their sites. A potential explanation for this is that adblocking extensions are a relative new phenomena on mobile devices. iOS for example allowed adblockers for Safari only with the introduction of iOS 11 in 2017. Furthermore, according to PageFair, currently mobile adblock usage is not widespread in Europe and North America, whereas in Asia mobile adblock usage has even overtaken desktop usage [36]. Therefore companies, especially those operating in Europe and North America, may have not yet seen the need to invest time and money to implement anti-adblocker scripts into their mobile sites. But with a wider adoption of adblockers on mobile devices, especially in Europe and North America, it could force website operators to implement anti-adblocking scripts also on the mobile version of their web-pages.

Overall, it may be said that mobile users using any adblocking solution are currently not being blocked of accessing content as often as desktop users, due to the fact that ad-blockers for mobile phones are rather new and less widespread on mobile devices in western countries. In Asia however, this situation could be different.

4.3.6 Video Advertisements

For the mobile web-pages the same behaviour as on desktop could be observed. Video advertisements on Youtube, Twitch or Facebook were blocked when using either of the adblocking extensions, but not when using Pi-Hole or Perfect Privacy. From this one could follow that adblocker extensions may be the best solution to block video on mobile devices advertisements as well. However, one has to keep in mind that most of the users do not use the mobile browser to access such portals. The mobile applications are much more popular, due to the additional capabilities and easier to use interface. In those applications none of the adblockers are able to block such video advertisements.
Since it is harder for mobile users to block such advertisements in the applications, companies such as Youtube or Twitch are using incentives such as ad free access, to sell their premium services to their customer. This may force users to buy such premium services in case they want to get rid of the advertisements, as no other solutions currently exist.

4.3.7 Data Plans and Money Saving

Next, we analyze the economical advantages a user could potentially have when using an adblocker to optimize data usage. It is not a big data hit when accessing the internet at home, where unlimited data plans are the norm, but it's a different story for users with limited data allowances for their mobile phones. Therefore we calculate how bandwidth savings of an adblocker could translate into real money savings for the end-user.
For this analysis the data by Cable.co.uk was used, which gathered and analyzed data from 6,313 mobile data plans in 230 countries between 23 October and 28 November 2018. The average cost of one 1GB was then calculated [33]. Figure 4.5 shows a world map with countries being colour-coded by the average price of one 1 GB of mobile data. The darker the country, the more expensive the data.

Nevertheless, it must be said that the true cost savings for each individual mobile user can be different. Factors such as roaming costs, different monthly limited data caps or special deals have a large impact in the prices the user pays in the end. The average data plans cost for this calculation should, however, be able to give a direction.

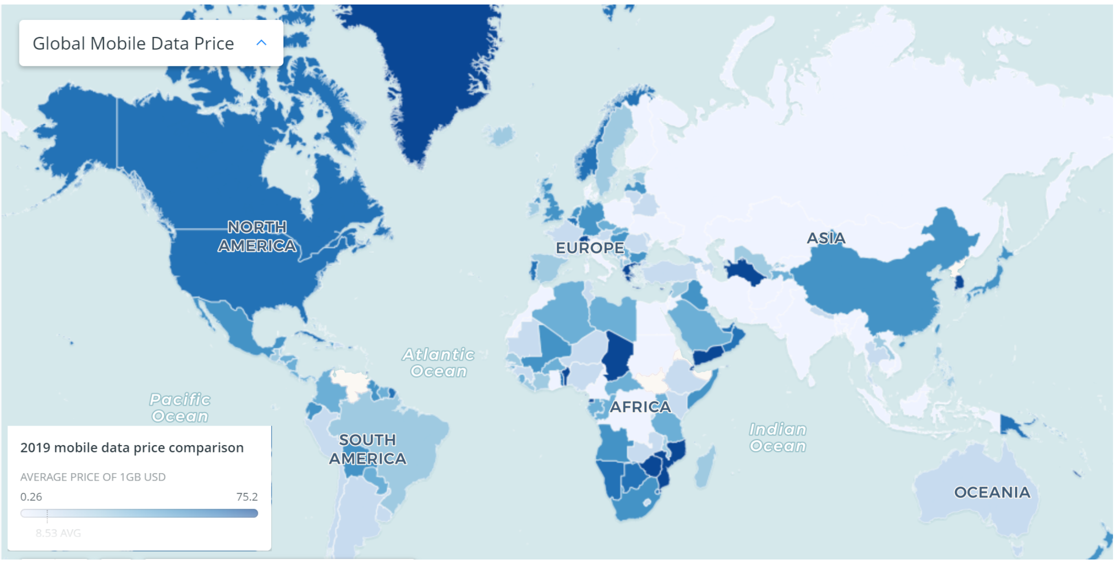


Figure 4.5: Mobile Data Price around the World [33]

From the list of 230 available 15 countries were selected (*cf.* Table 4.4). Three countries from each of these regions; North America, South America, Europe, Asia and Africa. Of these regions the respective most expensive, the cheapest country as well as an average one have been selected.

Table 4.4: Average Price of 1GB Data in 15 selected Countries

Country	**Average Price of 1GB (USD)**	**Continental Region**
Switzerland	$20,22	Europe
Germany	$6,96	Europe
Ukraine	$0,51	Europe
USA	$12,37	North America
Mexico	$7,38	North America
Guatemala	$4,53	North America
Falkland Island	$47,39	South America
Brazil	$3,50	South America
Chile	$1,87	South America
South Korea	$15,12	Asia
India	$0,26	Asia
Singapore	$3,67	Asia
Zimbabwe	$75,20	Africa
South Africa	$7,19	Africa
Rwanda	$0,56	Africa

Now based on the bandwidth saving results of the previous test, it has been assumed that about a 24% saving can be expected when browsing the internet with Pi-Hole. Out of the numbers from Cable.co it was then calculated how the bandwidth savings translate into real money savings. Figure 4.6 provides an overview of the savings with noticeable differences apparent.

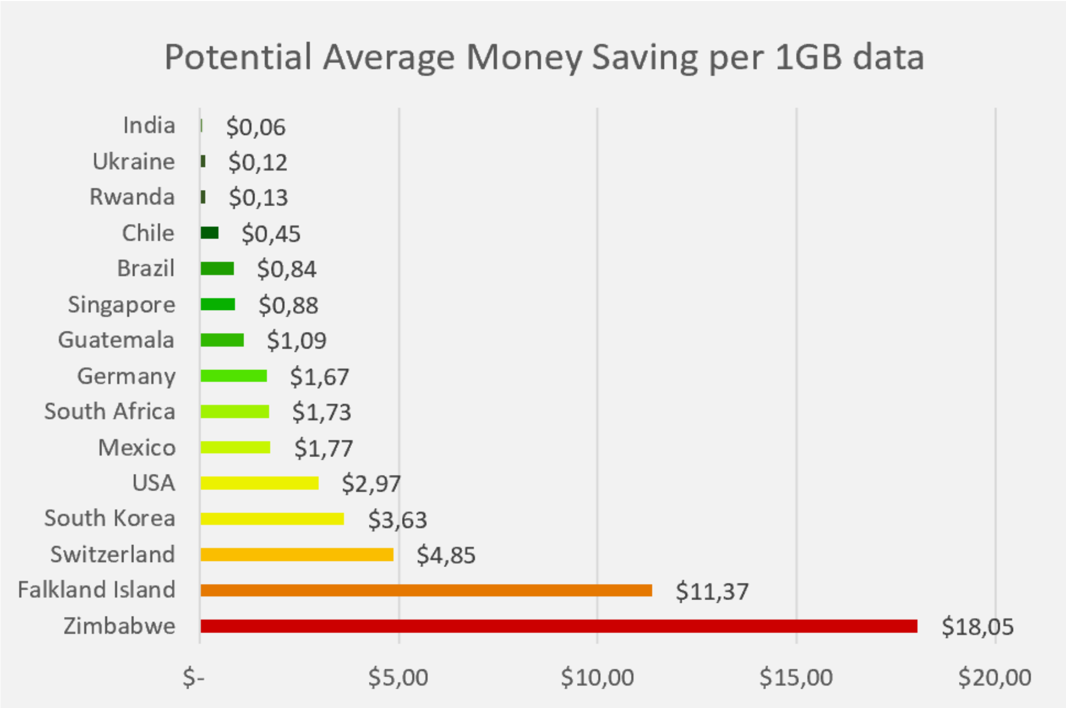


Figure 4.6: Potential Average Money Savings per 1 GB Data usage

Figure 4.6 shows the potential savings users with limited data plas could have in the 15 selected countries. Zimbabwe is the most expensive country for data, where a gigabyte averages \$75.20, therefore the potential savings were with \$18.05 the highest, while the savings in India were the lowest with \$0.006. In Switzerland, due to the high salaries and therefore high prices of data, an user could potentially save up to \$4.85 per 1 GB of data only with the use of a adblocker. In these cases, higher data usage of several GBs can lead to high savings for the user. For the rest, the savings are between \$0.5 to around \$3, while the median is \$1.67 overall.

Generally speaking in countries with higher salaries or bad broadband infrastructure higher money savings can be expected. Ranging from \$2.97 in the USA to \$18.06 in Zimbabwe. Additionally, in cases where a user has not closed a contract for cellular data usage or roaming is used, the money savings can be substantially higher as consumers then have to pay a high amount of money per MB. However, in countries with good broadband infrastructure or lower salaries the savings are rather low. In this case the potential money savings for the end user is ranging from \$0.006 in India to about \$1.77 in Mexico, while the average for such countries is around \$1.50. Nevertheless, 0.06\$ in India or \$0.84 in Brazil may seem low, but it needs also to be factored in that the average salary is much lower in those countries than for example in Germany.

4.4 Discussion

The results showed, as seen in Figure 4.7, that the use of an adblocker can have significant positive impacts on not only the performance but also on the total amount of bandwidth used. Overall Pi-Hole proved to be the best solution with average performance improvements as well as bandwidth savings of over 20%. Also content was not blocked when accessing some sites with Pi-Hole, which were previously blocked when those sites were accessed with Adblock Plus or uBlock Origin. But still a large disadvantage for Pi-Hole was the fact that it can not block video advertisements on many of the most popular portals. Additionally it does sometimes break web-pages, since it does not use CSS to hide the advertisement elements that are being blocked. To resolve these issues it can be useful to use Pi-Hole in conjunction with a adblocker extension. It showed that the additional usage of the extension did not have any noticeable negative impact.

(a) Desktop (b) Mobile

Figure 4.7: Performance and Bandwidth Usage Comparison of Adblocker Tools

Interestingly Perfect Privacy did consume by far the most amount data, due to the fact that VPNs encrypt traffic. Still it did perform better than Vanilla in regards to performance. Furthermore, noticeable differences between the adblocker extensions was able to be observed. Adblock extension user are therefore advised to select one, which has proven to be fast and efficient.

On the other hand noticeable differences between mobile browsing and desktop browsing could not be observed. Hence meaning that the amount of ads and the size of ads displayed on mobile web-pages and desktop web-pages is similar. Overall every adblocking solution has some advantages and disadvantages. Depending on the use-case one has to choose, which is best suitable for it. Table 4.5 shows the advantages as well as disadvantages for each of these solutions.

Solution	Advatanges	Disadvantages
Adblock Plus	*(i)* Available on almost all browsers and operating systems *(ii)* Easy installation and configuration	*(i)* Compared to the other adblocker extensions, it was performing worse in regards to performance and bandwidth usage *(ii)* The by default activated function Acceptable Ads can noticeably reduce the benefits in regards to performance and bandwidth usage
Pi-Hole	*(i)* Bandwidth usage and performance improvements were the best out of all adblockers *(ii)* Can also be used on non traditional devices such as Smart TVs *(iii)* Not as often blocked from the content as other adblocking tools	*(i)* Not able to block video advertisements on popular portals (e.g., Youtube, Twitch) *(ii)* Can cause pages to break, due to the fact that no CSS or JavaScript is used to hide the empty ad spaces
uBlock Origin	*(i)* Better performance and less bandwidth usage than Adblock Plus *(ii)* Easy installation and configuration	*(i)* uBlock is only available for desktop PC users *(ii)* Some sites block the content when using uBlock
Perfect Privacy	*(i)* Available on PC as well as Mobile *(ii)* As a VPN it provides many additional advantages such as additional security and privacy *(iii)* Not as often blocked from the content as other adblocking tools	*(i)* Very high bandwidth consumption due to encryption *(ii)* Since it is a premium service, monthly costs *(iii)* Can cause pages to break, due to the fact that no CSS or JavaScript is used to hide the empty ad spaces

Table 4.5: Overview of Adblockers Tools

4.4.1 Limitations

It is worth noting that the analysis and the comparison of the different adblocking extensions was based on the assumption that the different adblocking tools block about the same amount of advertisements on the different web-pages. We did not investigate whether an adblocker is performing better on particular pages by blocking more advertisements. The effectiveness of adblockers, however, can vary significantly [11], which in the end is one the most important factor when deciding which adblocker to use. Thus, no final judgment can be made based on the data obtained in this theses on which adblocker is universally speaking the best one to choose.

Moreover, the user behaviour simulations were based on data from SimilarWeb. We then tried to re-enact the two different user behaviors based on those metrics. But for an exact analysis real user data for example from ISP providers would be needed. The results, however, should still be able to give an indication on what users can expect when using those adblocking solutions.

Furthermore we did not investigate the impact of advertisements in mobile applications, due to the limitation of the simulator tool. Pi-Hole, for example, can also block some advertisements in mobile application, thanks to the technique used to block ads. Therefore it would have been a major advantage over the other tools in regards to bandwidth and performance analysis if application usage would have been a part of the simulation and analysis.

Chapter 5

Summary and Conclusions

Today many different adblocking solutions with different techniques to block ads on the internet exist. Users use these adblockers to get rid of annoying and unwanted ads on the internet. The goal of this thesis was to look at the effects of those adblocking tools from a user-experience and economical point of view. First, a simulator tool, that is able to generate internet traffic, was implemented with the help of different libraries such as PyChromeDevTools and psutil. Second with data provided from SimilarWeb desktop user behavior and mobile user behavior was simulated by opening 100 web-pages automatically with the simulator tool, which collected data simultaneously during each of these tests with the different adblockers. An in depth analysis was then conducted on the obtained data.

The main observations are that the blocking of unwanted ads through the use of adblockers leads to noticeable improvements for the end-user not only by loading the web-pages faster, but also by reducing the actual data consumption significantly. In fact, all solutions did improve the loading speed of web-pages. Performance improvements of 3% from a VPN solution like Perfect Privacy on desktop web-pages to faster performance improvements of 24% with Pi-Hole on mobile web-pages were able to be obtained. Noticeable reduction of bandwidth usage was also observed with the adblocker tools, except with Perfect Privacy, which increased the bandwidth usage by almost 55%. This is due to the encryption of data by VPNs, which creates additional overhead in the network. Pi-Hole once again was the best solution by reducing the total amount of bandwidth used on desktop web-pages by 21% and on mobile by 24%.

Major differences between desktop and mobile surfing in regards to the amount of advertisements shown on those platforms could not be observed based on the results. Thus it can be assumed that the number and size of advertising on both platforms is about the same. Moreover, it was able to show that the loading of unwanted ads can have real economical impacts on users, that use limited data plans when browsing the internet. Money savings ranging from $0.06 per GB in India, a country with very cheap data plans up to $18.05 per GB in a country like Zimbabwe, which has bad broadband infrastructure, can be achieved.

Nevertheless, one has to keep in mind that for many online content creator advertising is one of the major if not the only source of income. At a certain point in time, when adblockers are so widespread, that the revenue from advertisements do not cover the operating costs, content creator may need to force users to pay for their content, use anti-adblock scripts like many online news papers already do or in the worst case, back completely from the business. Therefore, it is still important that also end-users understand that viewing those advertisements allows them to consume the content for free. However, website operator must also note that overflowing their web-pages with advertisements for more revenue will make the usage adblockers even more popular.

5.1 Future Works

This thesis takes a first step towards understanding how end-users profit from adblockers by not only by getting rid of the annoying advertisements but also by providing other benefits such as improved loading speed or less data usage, which can result also in economical benefits. Yet, it does also open up new questions.

In this thesis the traffic was generated and then analyzed with the data SimilarWeb provided on the most visited websites in Switzerland. In a further step for more advanced analyses based on actual user data like for example data from ISP providers could be used. Furthermore it can be investigated what the impact of adblockers are not only on bandwidth usage and performance but also on CPU usage or the total amount of HTTP requests.

Additionally, with the use of Artificial Intelligence, it could be possible to modify the simulator tool in such a way that the internet surfing is done randomly on the internet and differences about the amount of advertisements displayed on web-pages are shown. But also in a further step the effectiveness differences between the adblockers tools could be analyzed in more detail, which are crucial for the end-user when deciding which adblocking solutions to use.

Bibliography

[1] Jasmine Enberg, What's Shaping the Digital Ad Market, 2019, [On-Line], https://www.emarketer.com/content/global-digital-ad-spending-2019, last visit 1 May 2019.

[2] R. Lee Zeff, B. Aronson: Advertising on the Internet, 2nd Edition, John Wiley & Sons, New York, NY, USA, 1999.

[3] M. Malloy, M. McNamara, A. Cahn, P. Barford: Ad Blockers: Global Prevalence and Impact, ACM Internet Measurement Conference 2016 (IMC),Santa Monica, USA , November 2016, pp. 119-125.

[4] Lic Daymette Montenegro Morales: The History of Advertising, CD de Monografias, Universidad de Matanzas, Via Blanca Km.3, Matanzas, Cuba, 2012.

[5] M. Tungate: Adland: A Global History of Advertising, 2nd Edition, Kogan Page, London, UK, 2013.

[6] OVK, Entwicklung ausgewählter Werbeformate, [On-line], 2017, http://www.ovk.de/online-werbung/daten-fakten/entwicklung-werbeformate.html, last visit 28 May, 2019.

[7] Cho Chang-Hoan, John Cheon: Why Do people avoid advertising on the Internet?, Journal of Advertising, Vol. 33, No. 4, December 2004, pp. 89-97.

[8] Pi-Hole, Pi-hole Network-wide Ad Blocking, 2019, [On-line], https://pi-hole.net/, last visit 28 May, 2019.

[9] AdblockPlus, Im Internet surfen ohne lästige Werbung, 2019, [On-line], https://adblockplus.org/de/, last visit 20 May, 2019.

[10] M. Ikram, M. Ali Kaafar; A first look at mobile Ad-Blocking apps, IEEE 16th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, October 2017, pp. [1-8].

[11] G. Kiran, K. Orestis, M. Michael: Ad-blocking: A Study on Performance, Privacy and Counter-measures, 9th International ACM Web Science Conference(ACM), Troy, New York, USA, June 2017, pp. 259-262

[12] The PageFair Team, 2016 Mobile Adblocking Report, 2016, [On-line] https://pagefair.com/blog/2016/mobile-adblocking-report/, last visit 10 July, 2019.

[13] TechoTepia, Participating in the Acceptable Ads Initiative - Ad Blocking, 2019, [On-line], https://www.techotopia.com/index.php/Participating_in_the_Acceptable_Ads_Initiative_-_Ad_Blocking#The_Acceptable_Ads_Controversy_.E2.80.93_A_Publisher.E2.80.99s_Friend_or_Foe.3F, last visit 28 May, 2019.

[14] Barbara Deleersnyder, Inge Geyskens, Katrijn Gielens, Marnik G. Dekimpe: How cannibalistic is the Internet channel? A study of the newspaper industry in the United Kingdom and The Netherlands, International Journal of Research in Marketing, Vol. 19, Issue 4, December 2002, pp. 337-348.

[15] Perfect Privacy, Perfect Privacy, 2019, [On-line], https://www.perfect-privacy.com/en/, last visit 28 July, 2019.

[16] Perfect Privacy,Get rid of ad-tracking and phishing sites with TrackStop, 2019, [On-line], https://www.perfect-privacy.com/en/features/trackstop, last visit 28 July, 2019.

[17] Python Software Foundation, What is Python? Executive Summary, 2019, [On-line], https://www.python.org/doc/essays/blurb/, last visit 2 July, 2019.

[18] Google, Google Chrome, 2019, [On-line], https://www.google.com/intl/de/chrome/, last visit 4 July, 2019.

[19] StatCounter, Browser Market Share Worldwide, 2019, [On-line], http://gs.statcounter.com/, last visit 2 July, 2019.

[20] ChromeDevTools, Chrome DevTools Protocol Viewer, 2019, [On-line], https://chromedevtools.github.io/devtools-protocol, last visit 2 July, 2019

[21] Python Software Foundation, Psutil 5.6.3, 2019, [On-line], https://pypi.org/project/psutil/, last visit 2 July, 2019.

[22] Python Software Foundation, Tkinter Python interface to Tcl/Tk, 2019, [On-line], https://docs.python.org/2/library/tkinter.html, last visit 4 July, 2019.

[23] SimilarWeb, Top Websites Ranking, 2019, [On-line], https://www.similarweb.com/top-websites/switzerland, last visit 3 August, 2019.

[24] Sandvine, The Global Internet Phenomena Report, October 2018, [On-line], https://www.sandvine.com/hubfs/downloads/phenomena/2018-phenomena-report.pdf, last visit 30 July, 2019.

[25] C. Javier Castro Peiia, J. Evans: Performance Evaluation of Software Virtual Private Networks (VPN), 25th Annual IEEE Conference on Local Computer Networks (LCN 2000), Tampa, FL, USA, November 2000, pp. 522-523.

[26] VPN University, Does a VPN use data? Can it help get around capped data plans?, 2017, [On-line], https://www.vpnuniversity.com/learn/does-a-vpn-use-data-can-a-vpn-unlock-data-caps, last visit 28 July, 2019.

[27] D. Lackovic, M. Tomic: Performance analysis of virtualized VPN endpoints, 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, 2017, pp. 466-471.

[28] R. Nithyanand, S. Khattak, M. Javed, N. Vallina Rodriguez, M. Falahrastegar, J. E. Powles, E. D. Cristofaro, H. Haddadi, S. J. Murdoch: Adblocking and counter blocking: A slice of the arms race, 6th USENIX Workshop on Free and Open Communications on the Internet (FOCI 16). Austin, TX, August 2016.

[29] Z. Shitong, H. Xunchao, Q. Zhiyun, S. Zubair, Y. Heng: Measuring and Disrupting Anti-Adblockers Using Differential Execution Analysis, Network and Distributed System Security Symposium Conference, Troy, New York, USA, June 2018.

[30] Wikipedia, SmiliarWeb, 2019, [On-line] `https://en.wikipedia.org/wiki/SimilarWeb`, last visit 2 August, 2019.

[31] E. Pujol, O. Hohlfeld, and A. Feldmann: Annoyed Users: Ads and Ad-Block Usage in the Wild; Internet Measurement Conference (IMC 2015), Tokyo, Japan, October 2015, pp. 93-106.

[32] Mariella Moon, Amazon, Google and Microsoft Escape Adblock Plus, for a Price, 2015, [On-line], `https://www.engadget.com/2015/02/03/amazon-google-microsoft-adblock-plus/`, last 31 July, 2019.

[33] Cable, Worldwide mobile data pricing: The cost of 1GB of mobile data in 230 countries, 2018, [On-line], `https://www.cable.co.uk/mobiles/worldwide-data-pricing/`, last visit 1 August, 2019.

[34] Victor Emmanuel, 5 significant benefits of Using a Virtual Private Network (VPN), 2017, [On-line], `https://www.lifehack.org/533452/5-significant-benefits-using-virtual-private-network-vpn`, last visit 4 August, 2019.

[35] Cisco, Cisco Visual Networking Index: Forecast and Trends, 2017–2022 White Paper, 2019, [On-line], `https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html`, last visit 8 August, 2019.

[36] Matthew Cortland, 2017 Adblock Report. 2017, [On-line], `https://pagefair.com/blog/2017/adblockreport/`, last visit 10 August, 2019.

Abbreviations

API	Application Programming Interface
CPU	Central Processing Unit
DNS	Domain Name System
GIF	Graphics Interchange Format
GB	Gigabyte
HTTP	Hypertext Transfer Protocol
IPsec	Internet Protocol Security
MB	Megabyte
ROI	Return on Investment
URL	Uniform Resource Locator
PC	Personal Computer
UI	User Interface
USD	United States Dollar
VPN	Virtual Private Network

List of Figures

List of Tables

Appendix A

Installation Guidelines

A.1 Tool

A.1.1 Description

To prepare for the of the tool you would need to setup a special Google Chrome instance. To use this tool, you must run an instance of Google Chrome with the remote-debugging option, like in the following example.

```
google-chrome –remote-debugging-port=9222
```

To add the remote debugging port to chrome, open the properties of a Google Chrome shortcut. Add then the above mentioned line to the end in the Target section.

A.1.2 Prerequisites and Installation

Very few dependencies must be satisfied: an updated Google-Chrome version and the python packages requests and websocket. Those should be installed by default. Furthermore following packages need to be installed.

- tkinter
- psutil
- threading
- PyChromeDevTools
- xlsxwriter
- xlutils

Secondly, a text file called simulator.txt needs to be created. The file should contain the list of URLs (*cf.* Appendix B), which should be opened by the simulator tool. Every URL entry in the file should be on a new line. Otherwise the tool will not work correctly. The file should be located in the same folder as the Python script.

A.1.3 Operation

To operate the simulator tool open first a instance of the Google Chrome browser with the remote debugging port configured. Now make sure that the simulator.txt files is located in the same folder. Next run the simulator using the follow command:

```
python simulator.py
```

This command should open the User Interface where the button to start the simulation is available. Now the simulation can be started. After all pages have been loaded, the results are shown on the User Interface as well as it will generate the results in a xls file (*i.e.,* Results.xls).

Appendix B

URLs

B.1 Desktop 100 Pages

The list of URLs used for the Desktop analysis can be found on following webpage `https://github.com/Lawand97/usersimulator`. The text file is called desktop.txt. To use it with the simulator tool, please change the filename to simulator.txt.

B.2 Mobile 100 Pages

The list of URLs used for the Mobile analysis can be found on following webpage `https://github.com/Lawand97/usersimulator`. The text file is called mobile.txt. To use it with the simulator tool, please change the filename to simulator.txt.

B.3 Source Code Simulator Tool

The source code of the simulator Tool can be found under following github repository `https://github.com/Lawand97/usersimulator`. Follow the Installation Guidelines to use the tool.

Appendix C

Contents of the CD

- Source files of the thesis
- PDF file of the thesis
- A german and english summary approximately 100 words
- The powerpoint slides of the mid and final presentation
- The results from the tests with all adblocker tools in an excel file
- Source code of the simulator tool