Improving the usability and security of low and high-latency anonymity networks

Killian Davitt

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Department of Computer Science

University College London

November 13, 2024

I, Killian Davitt, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

Useable software is critical for gathering a broad base of anonymity system users and enhancing anonymity guarantees. This thesis presents four projects that enhance the usability and security of low and high-latency anonymity systems and further this cause.

Firstly, the thesis assesses the usability of mixnet anonymity networks where high latency is required and can often affect usability significantly. Specifically, a user study is conducted, evaluating the tolerable levels of delay for users. The thesis argues that collaborative work is one of the most frustrating tasks for Mixnet users. A user study is constructed that simulates and evaluates this scenario, where a user answers questions with a second simulated user, intentionally designed to induce conflict.

The second project concerns "HTTPS-Only modes", web browser modes that enforce HTTPS connection and discourage users from loading non-HTTPS websites. Their use in Tor Browser is specifically considered, and they are analysed in anonymous settings. A qualitative survey of Tor Browser experts is performed, and Tor users' risk from unencrypted HTTP connections is explored.

Continuing the improvement of HTTPS-Only modes, new warning pages were designed, leveraging the previous survey and existing literature. These pages attempted to provide greater context to Tor Browser users and warn them of the risks they may face from non-HTTPS connections over Tor. A large-scale survey was conducted, and the effectiveness of these new warning pages was evaluated.

In the final chapter, the HSTS mechanism is studied. While normally an effective security mechanism, it is also used to track users and thus was disabled in Tor Browser. This project proposes a novel approach to share HSTS data among Tor Browser users. Specifically designed for Tor Browser, this protocol enables the use of HSTS to enhance user anonymity without tracking potential.

Impact Statement

The work in this thesis primarily concerns the ability of users to effectively make use of anonymity networks in order to conduct online activities anonymously. This work therefore has a direct social benefit in allowing users to enhance their online privacy. The property of online anonymity is highly desirable for at risk individuals like whistle-blowers or civil rights activists across the world, but also for regular users day to day use. Chapters 5 and 6 produce results that could be immediately implemented into the popular Tor Browser anonymity software that would immediately provide benefits to users.

Chapters 4 and 5 together created a new warning HTTPS-Only mode warning page that was shown to be statistically significantly improved over the current version of the warning. This new warning could be implemented into Tor Browser immediately. This work was completed with some input from the Tor Project and it is clear that a newly designed warning page is desirable for Tor Browser, this research has strong potential to fulfil this goal.

The research described in chapter 6 resulted in a publication at a leading privacy enhancing technologies conference giving it high visibility in the research community.

Chapter 3 is a more theoretical research work that primarily aims to direct the course of future work in the area. The chapter opens up the opportunity for a wide variety of future work into optimising specific applications for high-latency mixnets.

Acknowledgements

I would like to express my gratitude to my supervisor, Steven Murdoch. His support throughout the PhD process enabled me to persevere, even when things didn't go to plan.

I would also like to thank my friend, colleague, and collaborator Dan Ristea, who I worked closely with. Dan has always been available to offer advice on my work, always going above and beyond.

Thank you to Daniel Hugenroth for his time reviewing the thesis and offering much needed advice.

Thank you to all of the 169 lunch team for adding some fun and community to my time at UCL, especially Dan, Sharad, Jay, Madeleine, Nikos, Kyle, and others.

Significant acknowledgment must also be given to Plan Burrito, which fuelled the writing of this thesis.

My utmost gratitude goes to Tina. Without her support and encouragement, this thesis never would have been finished.

Contents

1 Introduction **19**

1.1 Thesis Contributions 19

1.2 Published Material 19

1.3 Author's Contribution 19

1.4 Research Ethics 20

1.5 Introduction 20

1.6 Thesis Structure 22

1.6.1 Anonymous Collaboration: How Does Delay Affect User Experience? 22

1.6.2 HTTPS-Only Modes: What Should They Do? 23

1.6.3 Creating New Warning Pages For HTTPS-Only Modes in Tor Browser 24

1.6.4 CoStrictTor: Bringing HSTS to Tor browser 24

2 Background **26**

2.1 Definitions of Anonymity 26

2.1.1 Measures of Anonymity 27

2.2 Anonymity Networks 29

2.2.1 DC-net 29

2.3 Mixnets 30

2.3.1 Threshold Mixes 31

2.3.2 Pool Mixes 31

2.3.3 Stop and Go Mix . . . 32
2.3.4 Dummy Packets . . . 32
2.3.5 Loopix . . . 33
2.3.6 Other Mixnet Techniques . . . 35
2.3.7 Anonymity of Mixnets . . . 36
2.4 Tor . . . 37
2.4.1 Entry Guards . . . 38
2.4.2 Exit Nodes & Malicious Behaviour . . . 38
2.4.3 Prevalence of Bad Tor Exit Nodes . . . 41
2.4.4 The Development of User Interfaces to Tor . . . 42
2.5 Tor Browser . . . 43
2.6 Usability of Tor and Tor Browser . . . 45
2.6.1 Tor Button, Vidalia, Privoxy . . . 46
2.6.2 Tor Browser Usability . . . 46
2.6.3 Tor Launcher . . . 49
2.6.4 General Design Conclusions . . . 51
2.7 HTTPS & SSL/TLS . . . 52
2.8 Methods of Automatically Upgrading to HTTPS . . . 53
2.9 Methods for Forcing HTTPS . . . 55
2.10 HSTS . . . 56
2.11 SSL Stripping Attacks . . . 57
2.11.1 SSL Stripping in Tor . . . 63
2.12 Other HTTPS Indicators . . . 64
2.13 HTTPS-Only Modes . . . 68
2.13.1 Tor Browser and HTTPS-Only Modes . . . 70
2.14 HTTPS Adoption . . . 70
2.14.1 Lets Encrypt . . . 72

3 Anonymous Collaboration: How Does Delay Affect User Experience? 74

3.1 Introduction . . . 74
3.2 Research Questions . . . 75

3.3 Structure of the Chapter 76
3.4 Related Work 76
3.4.1 Studies on Text Editing 76
3.4.2 Delay's Impact in Other Scenarios 78
3.4.3 Effect of Delay is Task Dependent 80
3.4.4 Users Come up with Strategies 80
3.4.5 Application Design Choices Mitigate Delay 81
3.5 Background 83
3.5.1 Typing Speed 83
3.5.2 Collaboration Architectures 83
3.5.3 CRDT's 84
3.5.4 Automerge 86
3.5.5 Appropriate Mixnet Delays 87
3.5.6 Potential Improvements in Deanonymisation Attacks 89
3.6 Design of the Study 91
3.6.1 The Simulated Second User 92
3.6.2 Collaboration Technology 93
3.6.3 Delay Levels 94
3.6.4 Out of Order Messages 96
3.6.5 The Questions 96
3.6.6 Collecting Results from Participants 97
3.6.7 Pilot Studies 99
3.6.8 Full Study Description 101
3.6.9 Recruitment 104
3.7 Results 104
3.7.1 Completion Times 104
3.7.2 Likert Scale Evaluation 108
3.8 Discussion 110
3.8.1 Excessive Delay Reduces the Speed of Collaboration 110
3.8.2 Completion Rate is not Reduced by Increasing Delay 113

3.8.3 Users Often Adopt New Strategies to Assist with Difficult Circumstances . . . 113
3.8.4 Recommendation to Mixnet Operators . . . 114
3.9 Limitations . . . 115
3.9.1 Learning Effects . . . 115
3.9.2 Scenario Does not Fully Mimic Reality . . . 115
3.9.3 Limited Abilities of Simulated User . . . 115
3.9.4 Limit on Upper Delay . . . 116
3.9.5 Does not Account for Larger Groups . . . 116
3.9.6 Applicable Only to Cases of Reasonably Synchronous Collaboration . . . 116
3.10 Summary . . . 116

4 HTTPS-Only Modes: what should they aim to do? 118
4.1 Introduction . . . 118
4.2 Research Questions . . . 120
4.3 Appropriate HTTPS Risk Models . . . 121
4.3.1 Users Should Expect HTTPS to be Available on Popular Websites . . . 121
4.4 Analysis of Current Warning Pages . . . 122
4.4.1 Google Chrome . . . 122
4.4.2 Brave . . . 123
4.4.3 Microsoft Edge . . . 123
4.4.4 Safari . . . 124
4.4.5 Firefox . . . 124
4.4.6 HTTPS Everywhere (EASE Mode) . . . 125
4.5 Survey design . . . 127
4.5.1 Recruitment . . . 127
4.5.2 Questions . . . 127
4.6 Results . . . 132
4.6.1 General HTTP Risks . . . 134

4.6.2 Does Tor Browser Come with Greater Risk? 138
4.6.3 Heuristic Measures of Risk 140
4.6.4 Other Interventions 142
4.6.5 Other Factors 144
4.6.6 Participants Views on Firefox (Current Tor Browser Warning Page) 144
4.6.7 HTTPS-Everywhere 145
4.6.8 Old Tor Browser Warning page 147
4.7 Discussion 149
4.7.1 Context is a Key Indicator of Risk from non-HTTPS Websites 149
4.7.2 Should Users be Encouraged not to Proceed 149
4.7.3 Improved Warning Pages 151
4.7.4 The Educational Utility of Warning Pages 154
4.7.5 Providing Agency to Users 155
4.8 Limitations 156
4.8.1 Unclear Sample 156
4.8.2 Lack of knowledge of HTTPS-Only modes 156
4.9 Summary 157

5 Creating New Warning Pages for HTTPS-Only Modes in Tor Browser 158
5.1 Introduction 158
5.2 Background 159
5.2.1 User Habituation to Warnings 160
5.2.2 Mental Models of HTTPS Connections 164
5.2.3 User Comprehension of Warnings 172
5.2.4 Warning Text & Style 174
5.2.5 Decision Retention 176
5.2.6 Summary of Conclusions from Background Literature . . . 177
5.3 Designing New Warning Pages 177
5.3.1 Popularity Warning 179
5.3.2 Context Warning 180

5.3.3 Tor Warning . . . 181
5.3.4 Readability . . . 182
5.4 Criteria for Proceeding Past the Warning . . . 184
5.5 Survey Design and Methodology . . . 186
5.5.1 Visiting a Bank: Do not Proceed . . . 188
5.5.2 Large Online Retailer: Do not Proceed . . . 189
5.5.3 Unknown News Website: Do Proceed . . . 189
5.5.4 Purchasing on a Small Independent Website: Do not Proceed 189
5.5.5 Menu of a Local Restaurant: Do Proceed . . . 189
5.5.6 Reputable News Website: Do not Proceed . . . 190
5.5.7 Tor and the Representative Sample . . . 190
5.5.8 Other Survey Design Decisions . . . 191
5.6 Results . . . 193
5.6.1 Did Participants Use HTTPS-Only Modes Before? . . . 193
5.6.2 Evaluation of New Warning Pages . . . 193
5.6.3 Warning Page Effectiveness by Scenario . . . 194
5.6.4 Warning Page Dwell Time . . . 197
5.6.5 Demographics . . . 197
5.7 Discussion . . . 198
5.7.1 Improvement in the Popularity Warning . . . 198
5.7.2 Lack of Improvement of Context and Tor Based Warnings . 199
5.7.3 Users are Hesitant to Proceed to Less Risky Websites . . . 200
5.7.4 The Reputable News Website Scenario . . . 201
5.7.5 Did the Warning Pages Promote Proceeding to Safe non-HTTPS Websites? . . . 201
5.7.6 Lack of Comprehension . . . 202
5.7.7 Actions for Web Browser Vendors . . . 202
5.8 Limitations . . . 203
5.8.1 Unusual Tor Network Imitation Scenario . . . 203
5.8.2 Results are Based on New, Untested Criteria . . . 203

5.8.3 Survey Gains a Limited Insight into how the Warning Effects User's Behaviour on Proceeding 204
5.8.4 Location Specific 204
5.9 Future work 204
5.9.1 Testing the Comprehension of HTTPS-Only Mode Warnings 204
5.9.2 Do Users Need to Proceed to Insecure Websites? 204
5.9.3 Users Need to Understand when to Proceed 205
5.9.4 Combining these Design Themes Could be Effective 205
5.10 Summary 205

6 CoStrictTor: Bringing HSTS to Tor browser 207
6.1 Introduction 207
6.2 Background 209
6.2.1 HSTS Supercookies 209
6.2.2 Tor Consensus 210
6.2.3 Tor Browser Deletes HSTS Flags 211
6.2.4 Differential Privacy 211
6.2.5 RAPPOR 213
6.3 Data Collection 216
6.3.1 Web Scraping 216
6.4 Protocol 219
6.4.1 Basic Overview 220
6.4.2 Threat Model 222
6.4.3 Protocol Design 223
6.4.4 Protocol Specification 231
6.5 Evaluation 234
6.5.1 Simulation Parameters 235
6.5.2 Results of the Simulation 238
6.6 Discussion 242
6.6.1 Limitations 243
6.6.2 Future Work 246

6.7 Related Work 247
6.7.1 Adaptations of RAPPOR to Unknown Dictionaries 247
6.7.2 Private Data Collection in Web Browsers 247
6.7.3 Other Additions to Tor Browser 249
6.8 Summary 249

7 Conclusion **250**
7.1 General Conclusions 250
7.2 Work Done in this Thesis 252
7.3 Future Work 254

Bibliography **256**

Appendices **292**

A High-latency Anonymity System Delay User Study Questions **292**

B HTTPS-Only Mode Scoping Survey Questions **318**

C Evaluating New Warning Pages Survey Questions **330**

List of Figures

2.1 Threshold mix 31

2.2 Pool mix layout 32

2.3 How dummy packets travel across the mixnet 33

2.4 How loop packets travel across the Loopix mixnet 35

2.5 A simple representation of an SSL Stripping Attack. In this setup, the user initially requests a HTTPS connection, the malicious server intercepts this request and does not reply to the user. Assuming the website does not support HTTPS, the user's browser automatically then makes an insecure HTTP request which is fulfilled by the malicious server. 58

2.6 The URL bar insecurity indicator in Google Chrome which is displayed whenever a non-HTTPS website is visited. 65

2.7 The mixed content warning previously displayed in Internet Explorer. It is clear that the warning provides no guidance for users who do not understand what HTTPS is. 66

3.1 The questions given to participants after each task session 103

3.2 Participants reported frustration level depending on the delay. An increasing level of frustration as the delay increases is shown. 109

3.3 Participant reported time effect from simulated user. As the delay increases, more participants believe the second user was slowing down their progress. 111

3.4 Participant reported strategy deviation: Slightly more participants had to change strategy in some way as the delay increased. 112

4.1 Chrome HTTPS-Only mode warning page 123
4.2 Edge HTTPS-Only mode warning page 125
4.3 Firefox HTTPS-Only mode warning page 126
4.4 HTTPS Everywhere (EASE Mode) warning page 126
4.5 Firefox HTTPS-Only mode warning page (version 96) 132
4.6 HTTPS Everywhere EASE mode warning page as shown in Tor Browse 133
4.7 Older Tor HTTPS-Only mode warning page (version 11) 133

5.1 Popularity warning 180
5.2 Context warning 182
5.3 Tor warning 183
5.4 Percentage of correct actions by scenario. The numbers shown here are not necessarily statistically significant. Please see Table 5.5 to see which pages and scenarios demonstrated statistically significant difference from the control task. 197

6.1 Protocol workflow 221
6.2 Protocol data is stored in two different epochs in order to model HSTS expiry times 225
6.3 Increasing filter size gives more accurate results until an optimal point past which accuracy declines 238
6.4 For the same ε, differing combinations of p and q produce negligible difference 240
6.5 Depending on the value of ε, the protocol can only a model a certain numbers of websites before false positives are too high. In this figure an excessive ratio of false positives to upgrades reduces the upgrade number to 0. 241

List of Tables

3.1 Demographics of the study participants 105

3.2 Education level of the study participants 105

3.3 Statistical testing of participants completion time versus the level of delay added. Completion times for each delay are tested against the control of the no assistance scenario. The lower delays of 1000ms and 4000ms show significant difference. The higher delays do not show significant difference. The Games-Howell test automatically accounts for multiple testing error and thus no additional correction needs to be done. 107

3.4 Mean completion times of the task. A general upward trend can be seen once delay is added, the highest level of delay results in a completion time almost as high as the no assistance scenario. 107

3.5 Size of effect for each delay level when compared with the no assistance scenario. 108

5.1 The SMOG reading level of the new warning pages (lower is better) 183

5.2 Participants reporting on whether they used HTTPS-Only modes before 193

5.3 Safety coding for each scenario 194

5.4 The effect on participants actions for each new warning page. Scenarios are grouped to show the overall effect. Only the popularity warning page shows significant change. 194

5.5 The effect on participants actions for each new warning page and each specific scenario, showing p-value and Holm–Bonferroni adjusted alpha values. Only two scenario-warning pairs are significant, both in the reputable news website scenario. 196

5.6 Dwell time results: there was no significant difference in the time spent on the warning pages compared with the control warning. (Correction was not used as none of the results were close to being significant.) 198

5.7 Demographic information of the participants 198

6.1 Frequency of HSTS expiry times 218

6.2 HSTS deployment by country for most popular 100 websites 219

Chapter 1

Introduction

1.1 Thesis Contributions

1.2 Published Material

Chapter 6 of this thesis is an extended version of a paper published as follows:

Davitt, K., Ristea, D., Murdoch, S. (2024). CoStricTor. Proceedings on Privacy Enhancing Technologies

1.3 Author's Contribution

Chapter 3 received generous input from Martin Kleppmann. His knowledge of collaborative data structures helped to guide the decision to use the Automerge library.

The work comprising chapters 3 and 4 received input from Matthew Finkel and Duncan Russell from the Tor Project. Discussions with these collaborators helped to motivate the projects and added insight to user interface design at the Tor Project.

The 'CoStricTor' published work which forms chapter 6 of this thesis was completed in equal collaboration with the second author, Dan Ristea. Developing the protocol, performing the protocol simulations, and writing the paper were all completed in equal collaboration.

Aside from the acknowledgements made above or explicitly referenced in the thesis, all other work featured in this thesis is my own.

1.4 Research Ethics

Three of the projects in this thesis involved research with human subjects, which required ethical approval from UCL's ethics committee. Chapter 3's user study gained ethical approval from the Computer Science Ethics Committee. Chapter 4's survey gained ethical approval from UCL's central ethics committee. Chapter 5's survey gained ethical approval from the computer science ethics committee.

Generally, the studies completed in this thesis did not require storing of personally identifiable information (PII) from participants. On the advice of the ethics committees, changes were made to ensure that PII was not accidentally collected, and if it was, processes were in place to delete it immediately. Other changes were made to comply with standard ethical requirements like ensuring participants were well informed and could exit the surveys at any time without consequence. Participants were provided with comprehensive information sheets for all of my work and these can be found in the appendices of this thesis.

1.5 Introduction

The use and utility of anonymity online is an important mechanism for maintaining individuals' civil liberties. This anonymity is often provided by Privacy Enhancing Technologies (PETs), in particular anonymity networks. Initially, these anonymity networks were only usable by experts; setting them up and ensuring they were functioning properly was a difficult task, and only those deeply immersed in the anonymity community could hope to engage them effectively.

One of the most important online anonymity principles is the anonymity set. An anonymity set is a group of individuals who share similar characteristics, making it difficult to distinguish and identify a specific person within the set. It is a fundamental concept in online anonymity, where a larger anonymity set enhances privacy by making individual identification more challenging. Additionally, as explained

by Mathewson and Dingledine [1], if anonymity systems are to be successful, the anonymity set must not only consist of a large number of people but also users from a wide variety of different backgrounds.

Conversely, if only a few individuals from a specific background use the system, it will be much easier to degrade their anonymity. If any recognisable trait is commonly linked to the users of the system, that association is likely to extend to you when you use the system. For instance, if the system has a reputation for being used by drug dealers, you may be profiled as a drug dealer when you use the system. Anonymity networks, therefore, must attract a broad base of users from all walks of life.

It is for this reason that usability has become a critical component of these anonymity systems. Enhancing the user-friendliness of these systems not only attracts more users and greater anonymity for all users. Many efforts in this domain focus on conventional usability enhancements, incorporating features such as clearer instructions, more informative warning text, and intuitively designed interfaces. Notably, systems like Tor Browser have already seen significant success with these improvements.

Other usability design improvements have made it easier for users to make themselves less distinguishable from one another while using these systems. Often, for example, this is achieved by reducing the number of configuration options available to users, making it more difficult for users to make their activity uniquely identifiable on the internet.

In this thesis, I study both high-latency and low-latency anonymity networks and critically contribute to improving their usability and security. For high-latency networks, I have selected the state-of-the-art mixnet design and performed a usability experiment to evaluate users' responses to delays caused by mixnets. For low-latency networks, the main candidate for study is the Tor network. I constructed a protocol to provide Tor users with greater security when browsing the web. I

also conducted two studies which attempted to enhance user agency in anonymity systems by improving warning pages. This work was performed on Tor Browser; however, its results also apply to Mix networks. I will now discuss how each project became a chapter of this thesis and how each project contributed to the main conclusion of the thesis.

The usability improvements that are typically made to anonymity systems largely originate from user suggestions. Users highlight the problems they have with anonymity systems and designers fix these. This leaves out a larger class of usability issues which are less predictable and are not generally noticed by users day to day, but can have major impacts if they are not addressed. This thesis identifies some of these issues, and then uses standard user centred security methodologies to address them. The thesis will show that it is worthwhile to continue to identify these less noticeable usability issues which is a process that works in tandem with fixing more typical usability issues that are highlighted by users themselves.

1.6 Thesis Structure

1.6.1 Anonymous Collaboration: How Does Delay Affect User Experience?

In my first project, I aim to assess the overall usability of collaborative systems over mixnet anonymity networks, focusing on the impact of added latency. Modern mixnet protocols intentionally introduce delays to heighten anonymity, with a direct correlation between increased delays and higher anonymity levels. However, this introduces a potential challenge: depending on the application, achieving high anonymity may render it impractical for users.

To address this, I conducted the first comprehensive user study in this chapter, evaluating the tolerable levels of delay for users. I argue that one of the most frustrating tasks for mixnet users is collaborative work with others. The application I constructed simulates this scenario, where a user answers questions with a second simulated user, intentionally designed to induce conflict.

The study's results offer practical guidance for mixnet operators, providing insights into determining the maximum tolerable latency before user frustration renders tasks impossible.

1.6.2 HTTPS-Only Modes: What Should They Do?

My next project takes a broader view and considers the broader dangers users face online. It is crucial to note that when considering anonymity, the user must be considered in a very broad context. It would be a mistake to focus entirely on network layer defences through mixnets, as it forgets the much broader attack surface that users face. It is for this reason that my next project concerns HTTPS. Robust standards like HTTPS web browsing are crucial when considering user anonymity. Without transport security through HTTPS, it is often trivial for an adversary to deanonymise a user. Simply eavesdropping on an unencrypted connection could be as bad as looking over the victim's shoulder and viewing their entire web browser activity. This says nothing of the potential harm that can also come from non-passive attacks made possible in the absence of HTTPS. Considering this, my second project concerns "HTTPS-Only modes", web browser modes which attempt to always enforce HTTPS connection and discourage the user from loading any non-HTTPS website. I specifically consider its use in Tor Browser [2], as it is perhaps the most common way users seek anonymity online, and I attempt to analyse the specific effects and considerations of HTTPS-only modes in anonymous settings. In this project, I explore how HTTPS-only modes are implemented across browsers and what benefits they give to users. I perform a qualitative survey of Tor Browser experts and draw out specific risks that Tor Browser users can face on the web from unencrypted HTTP connections. My study informs the future design of HTTPS-Only mode warning pages where, in the past, Tor Browser's warnings have always been directly copied from Firefox.

1.6.3 Creating New Warning Pages For HTTPS-Only Modes in Tor Browser

Building upon the findings of my prior project, I extended my investigation into HTTPS-Only mode warning pages within Tor Browser. Leveraging insights from the previous survey and existing literature on SSL warnings in web browsers, I created new warning pages. These pages attempted to provide greater context to Tor Browser users and warn them of the risks they may face from non-HTTPS connections over Tor. I devised new criteria to guide users on when to proceed past these warnings. I conducted a large-scale survey using these criteria and evaluated the effectiveness of these newly designed warning pages.

My results show a small, but statistically significant improvement in user responses to one of the new warning pages. Additionally, a significant amount of qualitative data was also gathered, showing users' various motivations for proceeding or not past the warning page. This work is the first to research users' behaviour in HTTPS-only modes when using Tor or otherwise.

1.6.4 CoStrictTor: Bringing HSTS to Tor browser

Tor Browser has been designed to enhance user's anonymity and privacy online. While the combination of Tor and Tor Browser provides a robust anonymous browsing experience, certain design decisions unintentionally compromise user security. In my final chapter, I address one such area: HSTS, or HTTP Strict Transport Security [3].

HSTS is a longstanding HTTP mechanism to mitigate HTTPS interception and man-in-the-middle attacks. However, as we'll discuss, its effectiveness in enhancing security comes with an unintended consequenceit has become a method for profiling users on the web. HSTS can be manipulated to store tracking flags in users' web browsers, allowing certain malicious websites to track users across multiple sites and compromise their anonymity significantly. This issue led Tor Browser to disable the use of HSTS, nullifying the effectiveness of these malicious trackers but also

sacrificing the security enhancement offered by HSTS.

In response, this project proposes a novel approach to share HSTS data among Tor Browser users. This allows the security benefits of HSTS to persist while eliminating its tracking potential. Specifically designed for Tor Browser, this protocol enables the use of HSTS to enhance user anonymity rather than compromise it.

Chapter 2

Background

2.1 Definitions of Anonymity

Before discussing the various methods of achieving anonymity online, it must first be defined what is actually meant by this. The broad definition from Pfitzmann [4] is taken as a starting point

"*Anonymity is the state of being not identifiable within a set of subjects, the anonymity set*"

This statement is non-specific but sets up the general concept of anonymity in online communications. To remain anonymous means that out of all the individuals using a system, it cannot be determined who is taking any particular action. The notion of the anonymity set is key when describing anonymity. System users cannot be truly anonymous; they can only be anonymous amongst all the other users. As an extreme example, if only one person is using an anonymity system, the system cannot, by definition, provide anonymity as there is only one possible answer for who is taking any particular action in the system.

So, as a result of this, if there is only one action taken in an anonymity system (e.g. one email is sent), by definition, there is no anonymity provided. Therefore, the definition of anonymity must be written in terms of multiple actions. This is

typically presented as the anonymity between two different actions.

More specifically, the concept of ‘unlinkability’ is a more specific and useful term. Unlinkability, in any anonymity system, refers to an inability to ‘link’ different items in the system, where an item could be an email sent anonymously, a payment transaction or perhaps a web page loading. Linking these items would be defined as determining if two emails in the system came from the same sender or were received by the same recipient.

Two more specific definitions are sender unlinkability and receiver unlinkability. A system can provide sender unlinkability, i.e. items cannot be linked to the same sender, but that same system may not provide receiver unlinkability. The postal system could be seen as an example of this: if no return address is printed on an envelope, it may not be known who sent the envelope, but it will always have the recipient’s details printed for it to be delivered.

Anonymity is a stronger definition. When referring to items in the system, an item is anonymous if it is not linkable to any user in the system. A system user is then anonymous if no item can be linked to them. If either of these terms applies to either all items in the system or to all users of the system, then the anonymity system provides anonymity. Thus, an anonymity system is anonymous if all of the actions taken in it cannot be linked back to any user. Definitions like this can vary in the literature, so it is key that I outline my own definitions as a background for explaining my work.

2.1.1 Measures of Anonymity

To create anonymous systems, it must be defined how anonymity is measured and what kind of results show that a system is anonymous. This can be particularly difficult to do as no one standard measure of anonymity can be applied to communication systems. As previously discussed, the theoretical labels of “unlinkability" and “anonymity" become much less useful after moving from providing true anonymity to providing probabilistic anonymity. To properly measure real-world anonymity, a

system's concrete parameters must be considered. Aside from more basic metrics like "number of people in the anonymity set", the most practical measurement of anonymity is usually the effectiveness of the best attack that can be performed on the system. By finding the most effective attack on the system, the level of anonymity offered is characterised as the performance of the attack. For example, an attack might produce an 80% probability of de-anonymisation given 100 hours of observation and with 2 hours of computation time and 10GB of memory requirements. In Murdoch's 2014 work [5], he begins at the most basic possible anonymity measures and builds up to more effective and more complex ones.

In the context of mix networks, at least, the most basic metric is to count the anonymity set of the user or the single packet which is being measured. This is not an effective measure, as the distribution of possible receivers of a packet to an adversary is unlikely to be uniform. Regardless, if an adversary does not know the true receiver of a packet, they still have certain confidences, e.g. that out of a system of 100 users, maybe 7 of them have increased likelihood of being the receiver for various external reasons perhaps. This totally breaks the model of the cardinality of an anonymity set as a reasonable measure of system security. If a system is particularly vulnerable to a disclosure attack, for example, it may have little hope of protecting the anonymity of its users, but this poor anonymity model may say otherwise. An improved general measure of anonymity is also proposed wherein both the anonymity set and prior probability distributions are considered. This proposes using the Shannon entropy of the connection. The following is used:

$$H(S) = -\sum_{i=1}^{N} p_i log_2(p_i)$$

p_i is the prior probability that i is the receiver of the packet. N is the cardinality of the anonymity set.

These various attempts at defining a method for evaluating anonymity are an important asset for this thesis. In order to evaluate how usable mixnet applications are, it

is essential to find out how much delay needs to be added. This can only be derived by measuring the level of anonymity provided by the system. It is not only important to understand how anonymity can be measured, but that the measurement itself is also a difficult and elusive process. There is no uniform and perfect measure of the effectiveness on an anonymity system and thus even if the anonymity system seems to measure up well against the current metrics, there is always the potential of flaws not just with the design of the system, but with the design of the evaluation of the system. The approach chosen in my own work is flawed in itself, attempting to evaluate anonymity based on discovery of the strongest available attack is clearly insufficient, as this assumes that the strongest attack which can be constructed and defended against is not weaker than any other unknown attack that can be constructed by an adversary. Much of the research on anonymity in the presence of traffic analysis must always take these kind of assumptions and considerations into account and indeed, the assumptions that are made when constructing new systems must not be forgotten.

2.2 Anonymity Networks

It is important and worthwhile to understand the methods of achieving anonymity and how they have evolved over time to today's systems. An anonymity network is a system or infrastructure designed to enhance the privacy and anonymity of users when accessing the internet. It achieves this by routing network traffic through a series of nodes or servers, making it more challenging to trace the communication back to the originating user. Here I will briefly describe the development of anonymity systems as well as their two main modern forms. For further information on anonymity systems, a recent SoK paper from Sasy *et al.* [6] can provide more details.

2.2.1 DC-net

One of the earliest forms of anonymity network or system is a Dining Cryptographer network, or DC-net [7] which is a primarily theoretical method for provably secure, untraceable anonymous communications.

In a DC-net, participants aim to communicate without revealing who is communicating with whom. Each participant continuously sends encrypted messages to all others, and only the intended recipient can decrypt a message. This constant exchange makes it impossible to determine who in the group is communicating with whom.

While DC-nets are theoretically powerful, practical limitations hinder their widespread use. For instance, in a DC-Net with n users, every user must participate in each protocol round, requiring each user to send a message to everyone else. This proves impractical and doesn't align with users' real-world expectations. Other practical anonymity networks relax these strict requirements, allowing users not to be online all the time. This trade-off sacrifices the provable anonymity of DC-nets for a probabilistic guarantee. An important point is that networks with variable user participation are susceptible to intersection attacks. Different anonymity networks employ various techniques to address this vulnerability, as we'll explore further.

2.3 Mixnets

The concept of a mix or mixnet has been around for some time, first introduced by Chaum [8]. Fundamentally, a mixnet is a set of servers that accept packets and forward them through other mixnet servers in an unpredictable manner, such that they reach the appropriate destination. However, it is difficult for outside observers to know the relationship between packet senders and receivers. Since their inception in the 1980's the technology has advanced somewhat. A number of different styles and designs of mixnet have been developed. As a variety of deanonymisation attacks were found on older mixnets, newer version were created that protected against these attacks. The newest forms of mixnets defend against a wide variety of attacks and implement numerous new techniques for maximising anonymity. Some historical mixnet architectures will be discussed briefly, and they will in turn assist with the explanation of modern mixnet designs.

2.3.1 Threshold Mixes

A threshold mix has a parameter t which is the number of packets the mix needs to receive until it outputs all the packets held in the mix in quick succession. The original Chaumian design for mixnets conforms to a threshold mix design. A threshold mix will receive packets over the network and add them to the buffer. When the number of packets in the buffer reaches the threshold number t the mixnet forwards all of the packets in the buffer to their intended recipients. Threshold mixes are basic designs, and the traffic analysis attacks discussed later are very effective against these designs, leading to some of the further design of the Pool and Stop and Go mixnets.

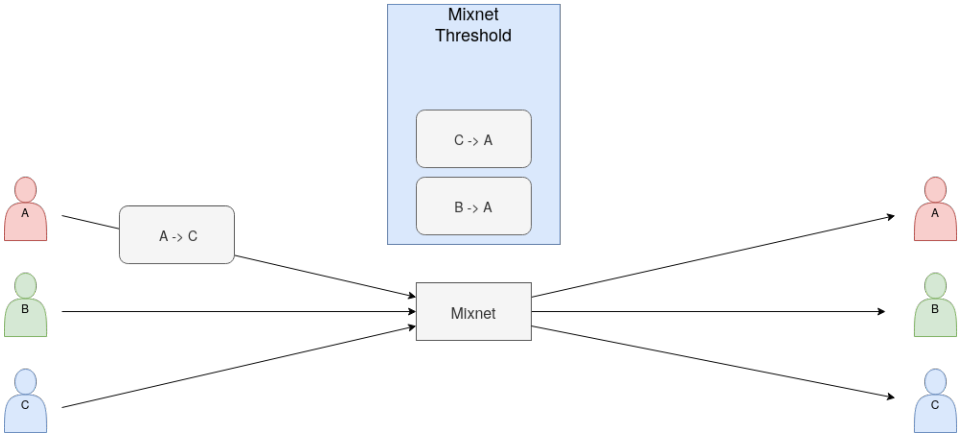


Figure 2.1: Threshold mix

2.3.2 Pool Mixes

Typically, a pool mix can be seen as a generalisation of a threshold mix, with more complex and varied conditions applied to the release of packets from the “pool" of acquired packets on the mixnode. Where a threshold mix stores packets up to n and then releases them, a pool mix could have a variable threshold number depending on factors within the mixnet, or could probabilistically hold some packets in its pool for longer while releasing others. Mixmaster, as I will discuss, is an example of a pool mix wherein, at fixed intervals, $x\%$ of the pool is randomly emptied [9]. The exact parameters of pool mix behaviour can be nuanced and seemingly arbitrary unless paired with attack effectiveness. In Serjantov’s 2003 work [10] some of these attacks are discussed concerning how effective they are over both different

mix types generally, as well as over different parameters to pool mix behaviour.

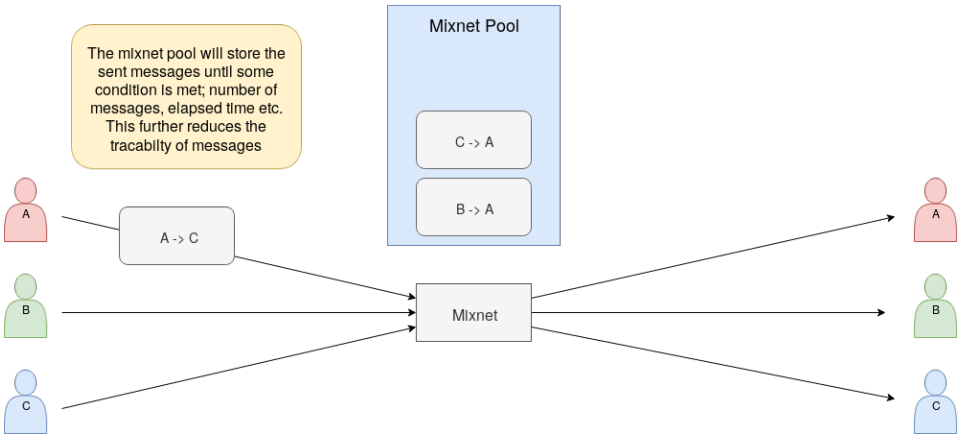


Figure 2.2: Pool mix layout

2.3.3 Stop and Go Mix

The stop-and-go Mix [11] is a different paradigm of mixing packets. Generally, stop-and-go mixes use probabilistic client-determined delays. Instead of thresholds or other pool methods, elements of the stop-and-go mix can be seen in Mixminion, Mixmaster and Loopix, which is discussed below. In these, the more traditional threshold or pool is omitted, and delays are added to packets themselves, typically by the initial sender of the packet. In Loopix, for example, when the sender constructs a packet, the sender samples several delay times and includes them in the packet structure, such that the mix nodes can use these numbers to delay that specific packet by that amount, as opposed to the mixnode deciding what the delay should be, in relation to the number of packets it has received.

2.3.4 Dummy Packets

In the most basic mixnet model, n users send packets to the mixnet in 1 round, and shortly after, n packets leave the mix and are sent to the appropriate recipients. One technique to reduce the effectiveness of attacks on this system is to introduce sending of dummy packets [12]. In this system, users who are “online" but are not currently sending packets, will send a redundant dummy packet to the mix, which is flagged in such a way so the mix knows to drop the packet. This results in a system where $n+m$ packets enter the mixnet and n leave and are sent to recipients,

which significantly increases the anonymity level provided by the system as the set of possible senders of any 1 packet is increased.

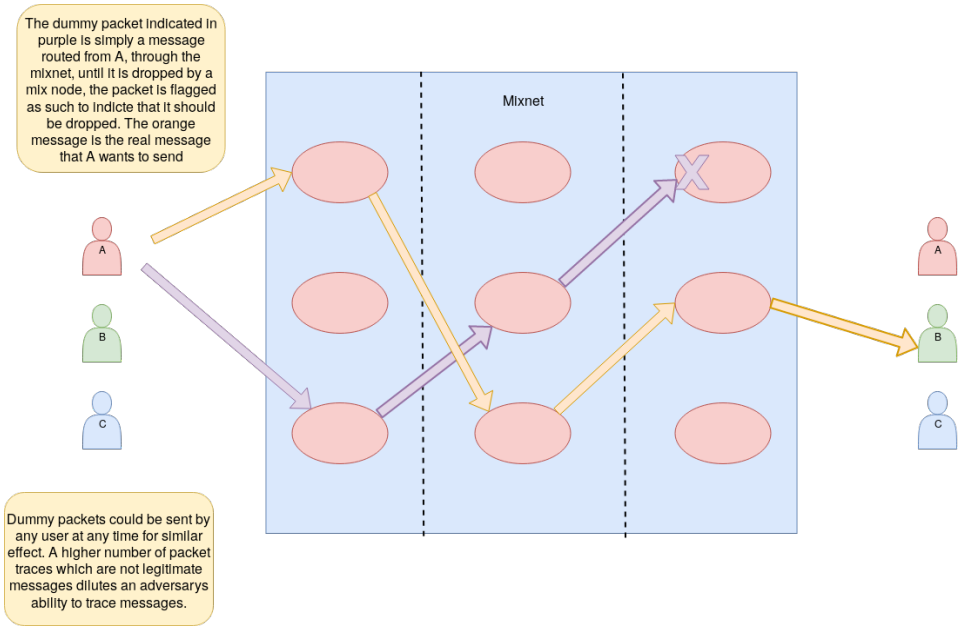


Figure 2.3: How dummy packets travel across the mixnet

2.3.5 Loopix

The Loopix anonymity system [13] is typically considered the most modern and advanced mixnet system design in existence. The system incorporates many of the features of previous mixnets, but with some new concepts as well.

The first key development of the Loopix system is the notion of using “Providers" to receive users’ messages. In this architecture, users do not interact directly with the mixnet, but instead, they have access to an inbox on a provider server. There may be many different provider servers and each one acts as a client of the mixnet. Users interact with the provider when they wish to send and receive messages through the anonymity network. As users come online and go offline, the provider server will typically remain online and can continue sending cover traffic to help increase anonymity. This crucial addition means that all nodes on the system can always be sending traffic to the mixnet, whether it is cover traffic or real traffic. As discussed, anonymity networks that deviate from the DC-net design are more vulnerable to disclosure attacks. Networks whose clients are always online and are always sending

messages are closer in design to the DC-net an thus are less vulnerable to attacks. This property is powerful by itself but is enhanced further by Loopix's next addition.

The baseline of the Loopix system can be seen as a stop-and-go mixnet, delays are added to individual messages in the network. Loopix gains much of its benefits from its slight deviation from this design. Packets are received in mixnodes as usual but, crucially, are held for individually sampled exponential delays. The original sender of the packet will generate the delays on the sender machine and include these delays in the packets (the mixnodes themselves do not sample delays). This is in strict contrast to pool and threshold mixes where the sending of packets from the mixnode depends on the state of the traffic overall and the arrival of other packets influence the path of this packet. As a result of this change, packet sending times in Loopix are independent. This provides the next notable addition of the Loopix system, the ability to model actions as a Poisson process. In contrast to other earlier mixnet systems, packets arriving at mix nodes will arrive according to the exponentially sampled delay which is independently sampled for each packet.

This crucially means that the rate of packets arriving at each mixnode is a Poisson process, and thus the overall rate of packet progression in the mixnet is a Poisson process. This is not just the case for real packets in the mixnet, cover traffic follows this process as well. This enables us to note that at all times, all clients of the mixnet are always sending traffic via a Poisson process. This is a key observation allowing us to more easily model processes in the system and assists us in evaluating the effectiveness of the system. The exponential parameter λ allows the mixnet administrator to uniformly adjust the delays introduced across the system, and effectively adjust the anonymity level provided by the mixnet in a way which would not be possible in other systems.

Another of the key additions to the Loopix system is an enhanced form of cover traffic, described as "loops"

This is a concept similar to typical dummy traffic however, instead of being dropped

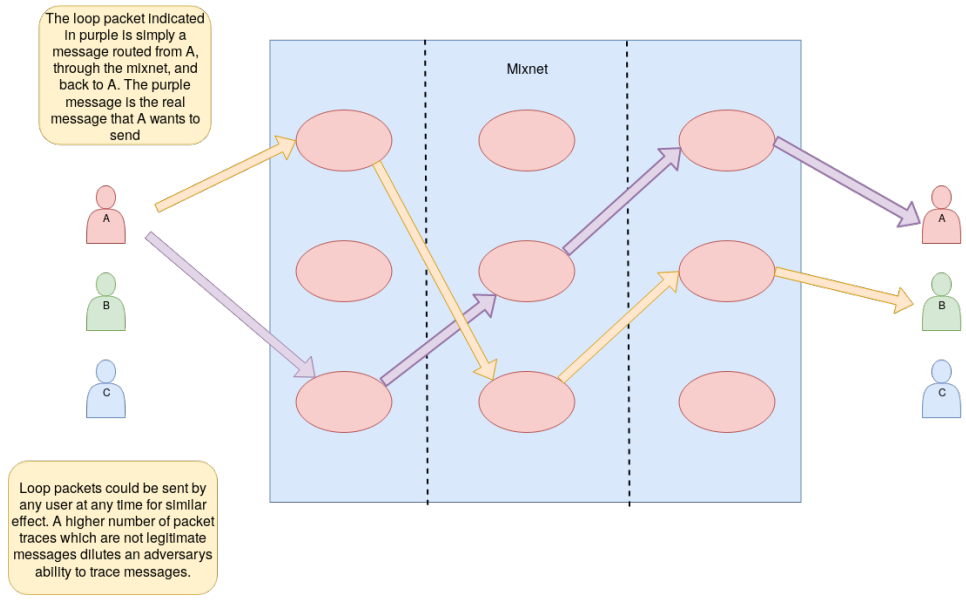


Figure 2.4: How loop packets travel across the Loopix mixnet

by one of the mix nodes, the loop traffic, traverses the full mixnode as normal and returns to the sender, thereby seeming like full and proper mix traffic to any adversary. Indeed, the loop traffic will even seem like real traffic to the mixnodes themselves and can prevent compromised nodes from assisting the adversary in this way.

2.3.6 Other Mixnet Techniques

As previously shown, each new iteration of and improvement upon mixnet technologies relies on new techniques to reduce an adversary's ability to deanonymise participants. A number of these interesting techniques which haven't been covered by the main concepts detailed above and will be reviewed here. Rollercoaster [14] is a novel attempt to solve an issue introduced when sending messages to larger groups using a mixnet. In a typical mixnet setup, if a user wishes to send the same message to multiple users, this will involve them sending n separate messages through the mixnet. This paradigm can be potentially very slow due to the mixnet delays being introduced n times, it is also potentially disastrous for anonymity as large noticeable message sending will increase the probability of a user being deanonymised by an adversary. Rollercoaster devises a method of mixnet broadcast to cascade the same message through users in a group. After receiving the message, users relay the

same message to other members of the group, thereby reducing the sending burden of the originating user. This is an interesting technique which is directly applicable to running collaborative group work on mix networks.

Another interesting technique is introduced in the Divide and Funnel scheme [15], this technique attempts to greatly increase the ability of mixnets to scale to a greater number of concurrent users without increasing latency. The key contribution of the system is to separate two key functions of nodes in the mixnet; computation, and mixing. In a traditional mixnet, all nodes in the network must perform asymmetric cryptography to 'remove a layer of the onion' and find the routing information for the packet, this computationally intensive operation restricts the ability of nodes to process large numbers of packets and so more nodes need to be introduced in order to scale the system, the downside of this is that with more nodes, the number of packets is divided amongst the nodes, and each packet is mixed with a small pool of other traffic. By definition, if fewer packets enter and leave a mix node in a shorter period of time, the anonymisation power of the mixnet is decreased. The techniques used in Divide and Funnel instead cascade a much larger number of packets through single mix nodes, after the initial cryptography has been performed by the larger cohort of 'compute nodes'. This key contribution is among the most modern and novel techniques for allowing mixnets to scale properly to real world implementations.

2.3.7 Anonymity of Mixnets

As was described earlier, any anonymity network that does not implement a dining cryptographers approach will always be fundamentally vulnerable to an intersection attack. This is therefore also true for mixnets. The anonymity of a mixnet is defined not as whether a user can be deanonymised immediately, but how long, and how much resources are necessary to deanonymise them. For this reason, there is no ideal limit on the settings of a mixnet system. The level of delay added by a mixnet can always be increased and it will be of further benefit. Much like other anonymity systems, mixnets are also vulnerable to the use of outside information to assist in

deanonymisation. Diaz [16] shows one example of how social network information can be used to assist in breaking users anonymity in Mix networks.

2.4 Tor

Having discussed the most prominent form of high-latency anonymity network in the mixnet, I will now move on to the most popular form of low-latency anonymity network which is Tor. Tor is one of the most popular methods of establishing anonymity online. First proposed by Dingledine *et al.* [17] and now run in practice by the Tor Project , Tor provides strong anonymity on the IP layer to traffic routed through its network. The Tor network is composed of many volunteer run *'relays'* or *'nodes'* which accept traffic from users and forwards it to other relays in the network. The protocol makes use of *Onion Routing* [18] to ensure a single relay cannot compromise the identity of users. When sending a packet through the Tor network, the packet is encrypted 3 times, in 3 layers. Each encryption is performed using the public key of the node the packet will be sent to, and this enables only the three intended relays to decrypt a layer of the 'onion'. Each relay only knows the source IP address and the destination IP address of the previous and next server in the path. The first relay knows the IP address of the user, and the middle relay, the middle relay knows the IP address of the other two relays in the path, and the last relay knows the IP address of the middle relay and the destination server. In theory, linking a user to the IP address they visited would require a compromise of all three relays.

Generally, Tor relays are desired to be located in many different geographic areas. This means that a user connecting to Tor can route their connection through many different jurisdictions and the probability of all three relays being compromised by the same organisation or individual remains low. Tor remains vulnerable to a *'Global Passive Adversary'*, an adversary that has large global influence and control. This adversary may be able to either monitor the network links of a large number of Tor relays, or who might have the capacity to directly compromise many Tor relays and thereby deanonymise users.

Tor's vulnerability to this type of attack is in contrast to most Mix networks. Unlike Tor, mixnets typically add additional delay to packets which renders them invulnerable to global passive adversaries. Tor's status as a 'low-latency' anonymity network has contributed to its ease of use and therefore its widespread popularity. Tor is a product used every day by millions of users, whereas mixnets are generally considered to have very few modern active services and with very low user counts.

2.4.1 Entry Guards

Given that the starting nodes in a Tor network path always know the IP address of a connecting user, it is considered more sensitive than other nodes. Nodes that can act as the first node in a path are given a special status known as *'Entry Guards'* or *'Guards'*. Guard nodes are appointed when there is additional confidence that the node is trusted and more reliable. This can result from the node being available on the network for an extended period and if it has significant bandwidth available for traffic. This additional trust level gives some additional confidence that the node is not compromised or corrupt, but it is not guaranteed or assumed. A Tor client typically selects a single Guard node to act as the entry point for all their communications for a set period, usually between one and two months. The appointment process for Guard nodes and the longer use of a single Guard node can significantly weaken adversaries' abilities to compromise users' identities. Suppose some proportion of Tor relays are compromised. In that case, constantly switching the entry node will increase the probability that one of the user's connections will go through a compromised node. In many cases, a single compromise of a user identity can be highly damaging; thus, protecting most of a user's communications is insufficient.

2.4.2 Exit Nodes & Malicious Behaviour

The last relay a user's traffic travels through is known as the *'exit node'*. This node forwards the traffic outside of the Tor network and to the destination it is intended for. Exit nodes remove the last layer of traffic encryption, so traffic leaves the exit node without any encryption. This can allow malicious exit nodes to attack the user's traffic, particularly if it is not already protected by TLS or another transport

level security mechanism. Given that the majority of Tor traffic is web traffic, the importance of widespread HTTPS use is paramount. As Tor nodes are run by volunteers, it is not difficult for adversaries to set up and run malicious or 'bad' exit nodes which attempt to use their position to attack users. Malicious exit nodes have the power to passively eavesdrop on all unencrypted traffic passing through them, as well as to perform active attacks by modifying the content of the traffic. This is why encrypted communications are encouraged over Tor, most notably by using TLS connections for web content.

Adversaries in this position have a lot of power to exploit users. Huber *et al.* [19] ran their own exit node and made observations on the traffic flowing through it. This ethically dubious yet informative experiment shows us the damage that can be done to Tor users' anonymity when using HTTP connections. The exit node was set up and advertised with a policy of only accepting HTTP connections. The authors ran the *dsniff* toolkit and recorded all HTTP traffic flowing through the server. 9 million HTTP requests were recorded over approximately one month. The content of the requests was not stored, only the header information. Requests were analysed by the domain being visited, the file type requested, and the user agent string reported by the user's browser. The authors conducted a domain analysis by categorising the domains into categories. Most requests belonged to Social Networking sites, Search engines, or file sharing. 7% of the sites being visited were classified as Social Networking sites. These sites were noteworthy to the authors as they are more likely to leak a large amount of information and potentially deanonymise the user very quickly. Using the user agent data, the authors find that potentially at least 78% of users were not using the *TorButton* extension. This extension previously was the main method of web browsing through Tor and provided an array of additional privacy protections which were considered essential for maintaining anonymity. Thus, these users were much more vulnerable to deanoymisation. The authors discuss some of the more general dangers that Tor users face when using unencrypted traffic. Firstly, information can be leaked directly from the HTTP request. Search queries can easily deanonmise users. The authors observed many

search queries in HTTP GET requests which had information about identity, location, nationality, etc. Social networks, as discussed, can also leak vast amounts of information. Requests can often include the user's identifier, which can also be their real name. The interactions on the platform can also leak information, leaking their social graph. Although not included in the authors' analysis, it is also envisioned that authentication cookies could easily be harvested and thereby compromise the user's entire account.

Script injection is the next possible attack described by the authors. If the exit node intercepts websites which request JavaScript, they can return altered or malicious code, which can deanonymise a user as shown by Abbott [20].

File replacement is the next risk the authors tackled. They identified that 1% of the requests in the data were requests for files with the potential for malware injection. Some examples are executable files (.exe), pdfs, Microsoft Office documents, or media files. All of these files have well-documented vulnerabilities that, if altered by an exit node, could run malware directly on the user's machine.

As a result of all of these potential risks for users, the authors suggest some mitigations. Detecting malicious exit nodes, and user education are mooted, however, here only on the authors' last suggestion is required: HTTPS. Many of the projects in this thesis focus on helping Tor users to adopt higher rates of HTTPS usage as this will completely mitigate any of the damages mentioned in this study.

This study was conducted in 2010, and since then, some critical problems have been solved. Many more prominent websites have switched to only allowing HTTPS connections, and one of the most significant domains in the authors' data, facebook.com, has long switched to only serving HTTPS content. Additionally, the rise of Tor Browser has resulted in most Tor users having identical user agent strings. Nevertheless, the study highlights the dangers of unencrypted transmissions over Tor and reinforces the need for ever-increasing rates of enforced HTTPS connections.

2.4.3 Prevalence of Bad Tor Exit Nodes

While it is difficult to accurately determine the number of actual bad exit nodes in the Tor network, many incidents have occured which indicate the presence of these nodes on the network. In leaks from Edward Snowden, GCHQ was shown to operate numerous exit nodes for surveillance purposes [21].

Mike Perry first found malicious exit nodes in 2006 [22]. In 2007, a security researcher allegedly collected sensitive information from government organisations around the world by using a malicious Tor exit node to sniff unencrypted SMTP email traffic [23]. In 2014, a malicious Tor exit node was found to be altering the binary files of software downloads [24]. It was later detected as part of a wider malicious operation deliberating distributing malware to Tor users [25].

Techniques have also been developed to identify bad exit nodes before attacks can occur. Chakravarty *et al.* [26, 27] pioneered a technique for detecting malicious exit nodes through honeypot credentials. Dummy login credentials were allowed to be captured by bad exit nodes through honeypot connections. Based on the number of attempts there was to use these fake credentials, and estimate could be made on the number of bad exit nodes encountered.

Winter *et al.* [28] designed a scanning setup which identifies malicious exit nodes. The system identifies malicious exit nodes as ones that provide invalid or mismatched x509 certificates on TLS connections, nodes that attempt to SSL strip, provide false SSH fingerprints, or provide inaccurate DNS information. A more complex system, HoneyConnector is also used to bait malicious exit nodes into extracting usernames and passwords from FTP and Email requests. These methods are effective in identifying malicious exit nodes. Some of the nodes discovered are clearly misconfiguration or benign errors; however, many attempted a form of malicious attack. Despite this great effort, however, it cannot be used to identify all malicious nodes. There are many situations where this method will not be effective. Nodes which only target particular domains for attack may not be easily identified, especially if the domains are uncommon and thus may not be tested. Attacks may

occur at different and limited times and may not be picked up by scanning efforts.

Sanatinia *et al.* [29] conducted work on identifying malicious Tor nodes through their behaviour when acting as an HSDir node. This specifically applies to Tor hidden services, which is not considered in this thesis; however, malicious nodes identified with this method could similarly attack non-hidden service connections.

In 2020, researcher Nusenu conducted extensive work analysing clusters of nodes appearing on the Tor network [30, 31, 32]. Using a variety of heuristic techniques, a very large cluster of bad exit nodes were found, which at one point formed over 23% of exit nodes on the Tor network. This attack coincided with, and was likely the cause of a large SSL stripping attack campaign against cryptocurrency websites which was leveraged to steal users' cryptocurrency [33].

It is clear that bad exit nodes operating on the Tor network is an ongoing problem that has persisted for many years, although new techniques have been developed to identify and block these nodes, they are not always effective and it has not yet been possible to eliminate this threat. One of the most important ways in which risks are addressed in Tor, including risk from bad exit nodes is through user interface design. To show this, I will next discuss how user interfaces have developed for Tor since its inception.

2.4.4 The Development of User Interfaces to Tor

User experience has been a core concern of the Tor project for many years. On its initial release in 2002, the software used to access the Tor network was basic and did not include any of the Web Browser interfaces that are commonly used today. Over time, additional software was created to enhance users' ability to access the Tor network easily [34]. *'Vidalia'* was developed to function as a graphical user interface for starting and stopping a Tor connection, and providing a local proxy where Tor intended traffic could be sent. *'Tor Button'* was developed as a Firefox addon which could activate or deactivate a connection to the local Tor proxy, allowing users to easily turn on and off a Tor connection on their browser. Tor

Button also featured many privacy oriented configurations, for example, when Tor Button was activated most browser plugins would be blocked to prevent them from accidentally deanonymising the user. Additionally, all cookies and cache data was temporarily removed to prevent sending of data over Tor which had previously been used in a non-anonymous browsing session. The functionality of Tor Button was eventually superseded by Tor Browser and instead of toggling in between Tor usage in Firefox, a users anonymous browsing would be confined purely to a separate browser instance. Tor Browser quickly became the default method of accessing Tor for web browsing and to this day, it remains the primary user facing software offered by the Tor project. Ultimately, it was decided to discontinue the Tor Button paradigm of using Tor [35] mainly due to the complexity of keeping separate states in the browser when it was toggled on and off, but also due to usability issues, e.g. users forgetting they had toggled Tor off and not turning it back on again when needed [34]. Tor's main offering had already changed from providing Tor access in Firefox to the separate Tor Browser. While Tor button was still available for Firefox and within Tor Browser itself, it was removed in 2011.

Vidalia was eventually replaced by the Tor launcher [36] and fulfilled essentially the same function of initiating the user's connection to the Tor network so that the browser could then be directed to connect through this Tor connection.

Eventually in 2021, Tor Launcher was effectively integrated into Tor Browser [37], allowing a more seamless connection and browsing setup. Tor launcher is now effectively a component of Tor Browser rather than a separate piece of software. Tor Browser is perhaps the most important development in Tor's user interface history and will be discussed next.

2.5 Tor Browser

In 2007, the Tor Project launched Tor Browser [36], a standalone web browser designed entirely to operate on the Tor network. Tor Browser implemented many of the privacy preserving configurations that were previously offered by Tor Button,

but allowed users to separate all of their anonymous browsing to a separate browser, and made it easier to avoid accidentally mixing anonymous and non-anonymous identities online. Tor Browser is derived entirely from Mozilla Firefox, and thus the developers can easily incorporate the latest useful Firefox updates to Tor Browser, while preserving the anonymity features it offers.

The Tor Browser design document [38] provides a wide ranging list of the specific privacy features that are implemented in Tor Browser to avoid accidental deanonymisation. This involves both disabling of features which could deanonymise the user without their intervention, but also takes account of situations where a user may use browser features to accidentally deanonymise themselves. An example of the former would be the browsers font fingerprinting protection. Given that adversaries can create fingerprints based on the fonts available on a users machine, many font queries can be used as a fingerprint. Therefore, Tor Browser places limits on the number of font queries that can occur from CSS, and also places a limit on the number of total fonts that can be used in a web page. This dramatically reduces the bits of entropy that can be used by an adversary to fingerprint a user. This is an example of an automatic intervention that does not depend on user interaction. Tor Browser also includes various protections that stop users from accidentally deanonymising themselves. While a highly knowledgeable user may know exactly what measures to take to prevent deanonymisation, Tor Browser takes the view that non expert users should be prevented as much as possible from taking obviously deanonymising actions. An example of this is disabling the geo location API in Tor Browser. Web sites which ask the user for their location will be automatically denied, and there is no possible way for users to allow the website access to this system. Normally in browsers such as Firefox, the user will always be presented with an option to allow or deny the request for location data. This prevents even an absent minded user from accidentally accepting a request for location data, when they might wish to remain anonymous.

The wide array of privacy enhancements as well as the isolation features ensuring all

activity with the Tor Browser is sent only over the Tor network makes Tor Browser the ideal way to access Tor, and it is the main software package offered by the Tor Project for this purpose. Tor Browser can be seen as a culmination of a longer history of user interface improvements to Tor. I will now discuss these developments in more detail and discuss the research work that has gone into evaluating the usability of these pieces of software.

2.6 Usability of Tor and Tor Browser

Collier [39, Page 99] details some of Tor's usability journey. Initially referring to the early days of Tor where technologies like Vidalia, Torbutton and Privoxy needed to be used, Collier opines 'For most of the everyday users that Tor aimed to attract, this was far too onerous' The intense expertise needed to set up a proper Tor connection both made it difficult, but also error prone. the Tor project worried that these issues would dramatically hinder the number of users who were not using Tor for criminal purposes. As a result of this, and with a desire to expand the anonymity set of Tor users, usability became a core aim of the Tor Project.

One of the biggest turning points for Tor usability was the release of Tor Browser by Murdoch in 2008. Collier describes this as: *"bringing Tor farther out of the bedrooms of computer enthusiasts and crypto-libertarians and into a much wider world of users."* Since then, Tor Browser has had usability and ease of use as one of its core functions. The Tor Project regularly performs user studies and surveys [40] usually in an attempt to understand users' issues with Tor Browser or to evaluate new features. One of the most recent examples is the March 2021 Tor Browser feedback survey [41]. The report gathered information about what users like and dislike about Tor Browser. As is common in other Tor surveys, participants reported speed as their main concern for Tor Browser usability.

Before the advent of Tor Browser, several methods were available to browse the internet using the Tor network. The Tor software connects to the network and provides a local proxy where traffic can be directed to. Most utilities assisting with

web browsing over Tor involve configuring the browser to access this proxy properly so that browser traffic is directly properly to the proxy and over the Tor network. The Vidalia software package [42] acted as a controller for the tor software, easily enabling users to turn tor on and off.

2.6.1 Tor Button, Vidalia, Privoxy

Clark *et al.* [43] conducted a usability analysis of some of the various methods of using the Tor network in 2007. Most prominently, Tor Button and Vidalia, and *'Privoxy'* which were endorsed by the Tor Project. The authors also considered *'TorPark'*, a web browser release specifically designed to provide Tor-enabled web browsing in a single downloadable package. The authors use a set of usability guidelines, for example, 'Users should be able to recognize, diagnose, and recover from non-critical errors.'. The analysis consists of applying these guidelines to the different Tor methods and checking if they apply whilst users complete various 'core tasks': installing Tor, configuring Tor, etc. This analysis showed a significant usability gap between the recommended Tor bundle (tor, Vidalia, Privoxy, Tor Button) and the TorPark browser. While not explicitly stated by the authors, it seems that the integrated experience of using one web browser software package is a vastly better experience for using Tor. I speculate that the high usability of TorPark may have inspired the subsequent development of Tor Browser, a software package similar to TorPark in that it contained all the necessary components of Tor in a single software package and allowed users to launch the browser and connect to Tor, rather than dealing with complex proxies. TorPark however lacked any official endorsement by the Tor Project and never become a widespread method of connecting to Tor.

2.6.2 Tor Browser Usability

The launch of Tor Browser slowly superseeded tools like Tor Button, Vidalia and Privoxy, but this did not signal the end of usability research in the Tor community. In 2012 and 2014, Norcie *et al.* [44, 45] conducted 2 user studies on the Tor Browser bundle, firstly identifying usability issues in the current setup and identifying solutions with recommendations to the Tor project. The second user study then tests

these fixes and finds that they effectively help the usability of Tor Browser. The primary issues found involve launch time, general browsing delay, and confusion about browser windows.

In 2018, Gallagher *et al.* [46] conducted a study which identified large gaps between the understanding of Tor between expert and non-expert groups. They argue that steps must be taken in the design of Tor Browser to prevent users from making errors which damage their anonymity.

Most recently, Viswanathan, in their 2022 Masters' thesis [47], completed a user survey of Tor Browser users. The survey explored different issues which were found while using Tor Browser. The main two issues documented were differential treatment by websites and general browser latency.

2.6.2.1 Informing users properly

Gallagher *et al.* [48] conducted a comprehensive mixed methods study of the usability of Tor Browser. The study was conducted over one week and monitored 19 non-expert users across that week as they used Tor Browser and other browsers. The initial collection stage involved installing a monitoring Python script on participants' computers. The script detected whenever (A): a Tor Browser session was closed, (B): A different browser was opened, or (C): the focus was switched between Tor Browser and another Browser. Whenever these events occurred, a questionnaire was launched which attempted to find any issues or negative events that may have caused the users to close Tor Browser or switch to another browser. The authors also conducted a semi-structured interview at the end of the experience and asked participants to complete a write-up summary of their experiences.

From this data, one of the common complaints of participants was that certain websites were not loading or working properly in Tor Browser. Another complaint was the latency of Tor and the extra time needed to wait for websites to load. Interestingly, in interviews, some participants expressed that they were initially annoyed by the latency but when informed that it was a consequence of Tor providing

anonymity, they were more willing to tolerate the delay. Secondly, the lack of some convenience features in Tor Browser was off-putting for some participants. Users already had bookmark information stored in other browsers which was not saved in Tor Browser. Additionally, Tor Browser does not support password saving, which some participants were frustrated by. That being said, other users understood that the lack of password saving was a security measure and thus were less annoyed by its absence. Some participants reported websites that blocked or hindered traffic from Tor Browser; although, overall, the number of occurrences was less than expected by the authors. Similarly, only 2 users reported issues due to geolocation. Some participants had issues with using DuckDuckGo as the default search engine. The authors also found that many participants did not fully appreciate the benefits of Tor. Some users were also confused by jargon-filled warning messages, which were not helpful to them. Users of Tor Browser may not be experts, and jargon-filled warning pages are often confusing and not actionable by users. Small errors like this, as well as other poor-quality UX, caused some participants to lose trust in Tor Browser and question its reliability and security.

It is also important to note participants' positive notes and remarks on Tor Browsers UI features. Participants enjoyed Tor Browser's feature of notifying them that window size changes could serve to deanonymise them. They also praised Tor Browser's feature of displaying the Tor circuit being used and which countries the nodes are located in. This likely gave users greater insight into how Tor works without burdening them with unnecessary jargon.

Overall, the authors proposed that a lot of frustration could be mitigated in Tor Browser by providing more information to users about the increased latency that might be felt, as well as other issues that may occur. The authors used the example of a certain type of PDF viewing failing in Tor Browser. Informing users that these issues may occur and that they are a trade-off when using Tor Browser can make them more tolerant of the issues when they occur and reduce frustration. Another fix proposed by the authors is to allow certain convenience features that would nor-

mally be disabled for privacy reasons to be enabled when Tor Browsers security slider is set to a lower level. The authors suggest a range of other improvements, including allowing users to specify which protections they are looking for in Tor Browser. This would allow certain features to be enabled or disabled to find an appropriate balance between security/privacy and ease of use. Crowdsourcing of websites that treat Tor traffic differently could be used to address these websites directly or warn users to expect problems. The authors propose other solutions to provide access to inaccessible content through Tor via caching schemes like the Wayback Machine. Crucially, the authors reinforced prior work, indicating the importance of using non-expert language that lay users can easily understand. Warning messages which avoid jargon and prior knowledge of Tor and Tor Browser are confusing and inaccessible to many users and can significantly degrade their experience.

Overall, this study showed a successful comprehensive usability study on Tor Browser, which provided actionable benefits and further lessons in the future design of anonymity systems.

2.6.3 Tor Launcher

Lee *et al.* [49] performed a usability evaluation of Tor Launcher, which is a piece of software used to configure Tor Browser's connection to the Tor network. Tor launcher has now been integrated into Tor Browser itself [50] although much of its functionality and interface remains the same. The Tor launcher ensures a proper connection to the Tor network; usually, this is a seamless process where users press a single button to connect. In many situations, however, direct connections to Tor relays are blocked, so Tor Bridges must be used[1]. This study explores users' experience of using Tor launcher to configure these Bridges and to navigate the different services they provide to enable connection to the Tor network. The authors found a number of issues that prevented users from properly configuring their Tor connection and thus could not use the browser. Most of the issues encountered involved

[1]Bridges are a different type of Tor node which must be used to access the Tor network when Tor's public entry nodes are censored. The IP address of Tor Bridges are generally kept more secret and thus it is more difficult to block access to them

terms that users were not familiar with, missing information in the menus, or situations where the launcher failed poorly (e.g., there was no timeout on connection failures, and users did not know how long to wait). The authors attempted to change the launcher's design to improve these issues but also adopted some design considerations to avoid creating any new issues from the changes. In particular, attempting to automate the functionality of Tor launcher to simplify the experience could result in undesirable outcomes. Some configurations, for example, may make known that the user is attempting to use Tor when it may be illegal to do so. Some configurations also cause an increased financial burden for the Tor project, and as such, they should not be automatically enabled.

The improvements made by the authors include reducing the amount of technical terms and the overall volume of text used. Increased clarification that some options are only necessary in heavily censored environments. Added auto-detection of proxies to avoid asking users questions they may not know the answer to. Provided more instruction on which type of bridge should be used. Added a final summary screen describing how the user is connecting to Tor.

To evaluate these changes, the authors conducted a quantitative lab study of 114 participants. The author's interface improvements showed significant improvements across the three different scenarios they tested (no blocking of Tor relays, blocking of Tor relays, and blocking of standard bridges). The completion rate (ability to connect) and connection and configuration time significantly improved. Regardless of these improvements, the authors do not necessarily suggest their improvements present the ideal interface. The real conclusion of the study is showing what types of general techniques are useful in improving the usability of a system like Tor/Tor Browser.

The first of the authors' conclusions is that designers should assume that users are not familiar with technical concepts; specific terms like Bridge, Proxy or Transport should not be required to use Tor, and thus using them in instructions is not helpful. Secondly, explaining to users why they must make certain decisions is use-

ful. Thirdly, leveraging users' own knowledge about their situation enables them to make better decisions about their configuration. Specifically, some users may have different risk tolerance levels, and allowing them to decide for themselves will result in a better experience for them. The user interface can ask users to make these choices directly instead of indirectly asking users about technical terms. For example, explaining different pluggable transports to users is time-consuming and difficult; however, since users are aware of their own risk tolerance, the interface could instead explain which pluggable transport to use for each level of risk.

Experiments by Fabian *et al.* [51, 52] evaluated the added delay for web browsing that results from using Tor to browse some popular websites. The median HTTP request latency was 5 times higher using Tor than otherwise, and DNS requests were almost 40 times slower over Tor. Tor has changed significantly since this 2010 study, so these figures are not provided as a current measure.

2.6.4 General Design Conclusions

The first steps taken by Tor's designers into improving Tor usability involved crafting better tools to access the Tor network. Initially this consisted of improving tools like Vidalia and Privoxy. These tools were eventually replaced by Tor Launcher and that in turn was ultimately folded into the Tor Browser itself. After some time, Tor could be accessed from one single software tool and a major usability hurdle was conquered. The more modern usability focuses of Tor, highlighted by multiple studies are on more specific areas. Latency is one issue frequently mentioned by study participants and has long been a problem with Tor, over the years latency issues have significantly improved although it still remains slower than traditional web browsing.

Another current usability issue if differential treatment of Tor traffic by websites, users frequently mention this as a problem, often users are blocked from websites when they are using Tor, or are forced to endlessly enter captcha forms which can be frustrating for users.

Finally, the most recent usability improvements in Tor are specifically tied to interface design itself. The integration of Tor launcher directly into Tor Browser was one such recent improvement. Users are no longer faced with a array of different Tor connections options before connecting and they are only given such options if a standard Tor connection fails [37]. This follows the tradition of removing as much extraneous complex information from users as possible to make the experience less confusing. Users are not faced with complex issues like Bridges, and obfuscation techniques unless necessary. These types of improvements continue, with recent improvements to tor circuit visualisation [53], vastly improved Bridge connection settings [54], introduction of an easier to use new identity button [55], and the improved security slider option to allow users to more easily determine their selected security level [56]. All of these options show the continuous updating of the Tor Browser user interface in order to give users the clearest and least confusing experience possible while dealing with a product which is fundamentally complicated. These usability improvements in Tor and its related software have been crucial in enhancing user's ability to stay safe and remain anonymous. A wide variety of issues have been addressed by the software to avoid common deanonymisation errors and other security issues.

2.7 HTTPS & SSL/TLS

This thesis presents further usability improvements to Tor Browser. These improvements are primarily centred around HTTPS and its use in Tor Browser. As such, I will now discuss HTTPS, the SSL Stripping Attack on HTTPS and the history of various web browser usability issues around HTTPS.

HTTPS is the secure form of the HTTP protocol. It provides robust security and privacy guarantees for users where HTTP does not. HTTPS effectively consists of the standard HTTP protocol wrapped within a SSL/TLS[2] connection. SSL provides confidentiality, integrity and authenticity to connections. Firstly, third party eaves-

[2]TLS previously was known as SSL. For simplicity, I will henceforth use SSL to refer to SSL to avoid confusion. So called 'SSL Stripping Attacks' which will soon be discussed, have never been referred to as 'TLS Stripping Attacks' and thus it is easier to refer only to SSL to avoid confusion.

droppers cannot view the encrypted content of the connection. Secondly, due to authenticated encryption, the user can immediately tell if the content of the connection has been tampered with during transit. Finally, the authentication infrastructure surrounding HTTPS: certificates, certificate authorities and public/private keys allow a user of HTTPS to determine the identity of whomever they are communicating with. This mitigates a wide variety of attacks where a man-in-the-middle attacker attempts to impersonate a legitimate web server.

The security guarantees offered by HTTPS make it highly desirable when using the internet. It's widespread adoption also means that a large majority of web connections are normally made using HTTPS and only a minority of popular websites continue to allow insecure HTTP connections.

2.8 Methods of Automatically Upgrading to HTTPS

Over the past years, to overcome these SSL attacks, and to generally enhance the usefulness of HTTPS, several methods have been used to encourage or force HTTPS use in web browsers. Doing so would allow users to benefit more from the protections offered by HTTPS and often this could be done with no disadvantage to the user. These techniques would only work when a website has deployed HTTPS, but also allows non-HTTPS connections. In this case, it is not difficult to voluntarily upgrade to HTTPS when it is available.

HTTPS Everywhere is a web browser extension [57, 58] developed by the EFF to increase web browser security by automatically upgrading HTTP connections to HTTPS. Aside from this primary use, HTTPS Everywhere also includes a configuration known as 'EASE Mode' (Encrypt All Sites Eligible). This mode was initially introduced in October 2014 [59] under the name 'http nowhere'. A proper warning page for this mode was not added until August 2018[3].

Sivakorn *et al.* [60] surveyed early HTTPS enforcing mechanisms. The set of differ-

[3]This effectively was the first HTTPS-Only mode available in web browsers, although it is unclear how widely used this mode was.

ent client-side features available in 2016 for web browsers is particularly relevant to this thesis. HTTPS Everywhere, 'Redirect to HTTPS', 'KB SSL enforcer', 'Smart HTTPS', 'HTTPtoHTTPS' and 'HTTPS by Default' were studied. As of 2023, HTTPS Everywhere [57] is the only one of these mechanisms still in use, and this last system is now being phased out in favour of modern integrated HTTPS-Only modes. Many of these featured mechanisms required greater complexity than the more modern HTTPS-only modes before the era of more ubiquitous HTTPS. In the past, many more websites were configured in a manner that would break functionality over HTTPS. This was particularly common for websites where HTTPS was only available in particular sections. A common example was Facebook, which only accepted HTTPS connections for login pages. This contrasts greatly with the modern web, where websites almost always support HTTPS in a binary fashion. It was necessary for the HTTPS Everywhere browser extension to comply with these older websites by employing a complex ruleset database. Entries into the dataset used regular expressions to specify which subdomains of the target URL should be automatically upgraded to HTTPS and which should be loaded via HTTP. The ruleset can also automatically mark specific cookies as 'secure'[4], ensuring they will not be produced for a website without an HTTPS connection. Rules were mostly crowdsourced; extension users could report rulesets to GitHub, where they would be included in the extension distribution. Mayer *et al*. [61] conducted a study on the quality of crowdsourced submissions and found that many submissions disagreed with each other, showing the complexity necessary for upgrade mechanisms at that time. HTTPS Everywhere was created by the EFF and has also had long term support from the Tor Project. HTTPS Everywhere has been included in Tor Browser for some time owing to the increased desire for secure connections to be used over Tor.

KB SSL Enforcer is another less complicated Chrome browser extension. This addon also attempts to upgrade via a rulset list however, it does not support more complex subdomain divisions and, according to the authors, does not handle specific

[4]Secure cookie in this case does not refer to the standardised secure cookie HTTP standard

configurations correctly.

Another extension surveyed by the authors, Smart HTTPS, upgrades URLs manually added to a list specified by the user. Which is more onerous for the user and limits the usefulness of this addon.

Finally, HTTPS By Default a more simplistic addon, upgrades all initial browser requests to HTTPS. This basic approach does not support websites that do not implement HTTPS and thus is likely of limited utility.

This limited selection of add-ons from Sivakorn et al. shows the initial stages of mass HTTPS adoption. Users were enabled to always request HTTPS connections for websites that supported them, this removed the obligation for users to specifically type "https://" for websites which did not specifically redirect to HTTPS connections. This removed some user burden and increased the overall coverage of HTTPS connections. Still, it did not assist users in avoiding insecure websites when they should expect HTTPS to be available nor, could have protected against more complex SSL Stripping attacks that attempted to degrade their security.

2.9 Methods for Forcing HTTPS

The above HTTPS adoption utilities were effective in increasing user security, however, HTTPS was never mandated by the addons, only automatically upgraded if it was available. A separate movement to force HTTPS upgrades without a silent fallback to insecure HTTP began with ForceHTTPS [62] in 2008. While this proposal does not contain explicit warning pages for non-HTTPS pages, it does combine automatic HTTPS upgrades with stronger and more explicit SSL warnings. This proposal embodied a concern for facing users with more explicit warnings for SSL and further adoption of HTTPS in order to prevent attacks from users, and this trend is what ultimately became HSTS as it appears in modern browsers.

2.10 HSTS

HTTP Strict Transport Security or HSTS is a protocol by which websites can instruct Internet browsers to only interact with them via a secure HTTPS connection [3]. Websites communicate their HSTS policy to browsers by setting the corresponding header in their HTTP response to a user request. When the browser encounters the HSTS header, it will store the HSTS status of the website and only allow future connections to be HTTPS. Forcing HTTPS connections prevents both passive eavesdropping attacks and active man-in-the-middle attacks like SSL Stripping. If a website known to have HSTS does not support HTTPS, the browser displays a warning page that does not allow the user to continue due to the high risk of an attack aiming to downgrade the security of the connection.

The HSTS header must include a *max-age* directive, which informs the user's browser how long to store the HSTS status. Websites can revoke their HSTS status by setting the expiry time in the header to 0 and waiting for any previously stated expiry time to elapse.

HSTS requires a user to first navigate to a website over a valid HTTPS connection for future connections to be protected by HSTS. To help mitigate this limitation, all major browsers (including Google Chrome, Chromium derivatives, and Mozilla Firefox) include the HSTS preload list [63]. The preload list is distributed with the web browser and updated when the web browser is updated. When initiating a connection, browsers will check both the saved HSTS list and the preload list. The preload list allows website administrators to mark their websites as having HSTS to all users even when they have not visited the website before. The recommended value for HSTS expiry is 2 years and the preload list requires this value for a website to be included [63].

In practice, the overall number of sites submitted to the preload list is low; therefore, it is of limited benefit. Many industries and types of websites were found to be poorly represented in the preload list [64]. Although the preload list has grown to approximately 190,000 entries at the time of writing, of which the Tor Browser

incorporates 138,000 entries, it scales poorly with the size of the public Internet. Of the top million websites by referring subnets [65], only 14,000 are included in the preload list.

HSTS is an actively recommended security feature for the web. Mozilla web security guidelines include HSTS as "*mandatory for all websites*", as HSTS has high value and low implementation difficulty [66].

In Section 6.3 of Chapter 6, I outline further data gathering I conducted to confirm the state of HSTS deployment on the web.

2.11 SSL Stripping Attacks

The protections offered by HTTPS are robust and can be difficult to get around for attackers. Adversaries can, however, take action to remove these protections in some instances. Although the authenticity guarantees provided by SSL mean an adversary cannot easily impersonate legitimate websites, an adversary can downgrade attempted HTTPS connections via a man-in-the-middle attack. This is known as SSL Stripping. Once SSL protection has been prevented from kicking in, the attacker can exploit the user's connection in numerous other ways; for example, this could be obtaining sensitive information from the web traffic (e.g. usernames and passwords), altering the web request content to present false information to the victim, amongst various other damaging attacks.

The simplest SSL Stripping attack is performed when the victim's web browser attempts a non-HTTPS connection to a website. The attacker intercepts the connection request, stopping it from reaching its intended location. The attacker then makes their own request to the website in question, and the website may mandate an HTTPS connection, which the attacker complies with. The attacker then forwards the website response back to the victim, thereby completing the victim's connection as intended but preventing any upgrade to HTTPS from taking place. While ordinarily, a non-HTTPS request to a website would be redirected to an HTTPS connection, the attacker sitting in the middle can complete an HTTPS connection with the web-

site and then serve the unencrypted content of the website back to the victim without any TLS connection having taken place; if successful, the user will not encounter an HTTPS connection, and they will assume the website did not mandate it. While this attack was first discussed in a general sense for all websites [67, 68], it has particular relevance for Tor Browser, where exit nodes are in an ideal position to deploy these attacks. These attacks provide even greater power to malicious exit nodes. Without this attack, they could only intercept data from voluntarily non-HTTPS connections (often for websites which do not support HTTPS). With this attack, however, far more websites can be victimised. Any website where a strong protection measure like HSTS[5] is not in place is vulnerable to this attack, vastly expanding the scope of bad exit nodes to deanonymise users of Tor.

SSL Stripping can also be performed by altering the targets of links on an unencrypted webpage. A victim may be browsing a website through a non-HTTPS connection; if the website contains links to other websites, the attacker can alter the address of those links. Links that were originally targeted at an HTTPS connection can be altered to drop the final 's'. Once the victim clicks this altered link, an SSL Stripping attack can be conducted as before, and the website is prevented from upgrading the victim's connection to HTTPS.

Figure 2.5: A simple representation of an SSL Stripping Attack. In this setup, the user initially requests a HTTPS connection, the malicious server intercepts this request and does not reply to the user. Assuming the website does not support HTTPS, the user's browser automatically then makes an insecure HTTP request which is fulfilled by the malicious server.

In these basic SSL Stripping attacks, if a user specifically requests an HTTPS connection to begin with, the attacker must respond differently. If a user types 'https' into their browser when visiting a website, the web browser attempts to initiate a secure connection. The attacker cannot respond satisfactorily to this request as they

[5] See Section 2.10 for an explanation of HSTS

will not have a suitable SSL certificate. In this case, the best response for an attacker is to drop the user's request and not respond. Most web browsers will, at this point, assume the website does not support HTTPS as it did not respond. The browser will then default back to a non-HTTPS connection and make the request again. Now, the attacker can resume the SSL Stripping attack as normal.

Protections which aim to defeat SSL Stripping have been developed, and one such protection is HSTS, which aims to mandate HTTPS connections for some URLs. SSLStrip Plus [69] is an improved version of the SSL Strip attack that can defeat HSTS protections on links clicked on from an unencrypted web page. This attack can sometimes require the use of DNS spoofing, which in some circumstances is more difficult for a man-in-the-middle actor to perform. In the case of Tor's malicious exit nodes, it is trivial. Tor Browser allows all DNS queries to be conducted by the Tor exit node being used. This is essential to preserve the user's anonymity. However, this allows the exit node to maliciously modify DNS responses as DNS continues to be mostly unauthenticated. The Tor Project has considered implementing DNS over HTTPS to mitigate this issue; however, it has currently decided not to implement the technology primarily due to the centralised nature of DNS over HTTPS resolvers [70].

SSL stripping attacks are not restricted only to Tor and have been seen to cause many security problems in regular browsing, including some high-profile attacks [71].

PrettyBadProxy [72] is a presentation of various other attacks that could be conducted by malicious proxies through which HTTPS connections were being made. Most of the vulnerabilities have now been fixed; however, some involve abusing the lack of secure cookie flags by web administrators, which can still be exploited. These attacks are distinct from SSL Stripping attacks. Still, they operate similarly, where a malicious man-in-the-middle can trick the browser into loading insecure HTTP pages and leaking users' personal data.

HProxy [73] is a technique developed by Nikiforakis to help defend against SSL

stripping attacks. Primarily, it leverages a user's browser history to determined characteristics of websites that have previously been visited. These characteristics are compared to the present version of the website which can help to indicate if a man-in-the-middle attack is in progress. By analysing the HTTPS status of different page components, the software can see if changes have been made to remove HTTPS and therefore, notify the browser that an attack is in progress. If for example, a password submission form usually has an HTTPS submission address and is found to no longer submit over HTTPS, an attack is likely, and the user can be notified. This technique has strong parallels with HSTS, where previous visitations to a website inform what should be expected on future visits. The technique is more powerful than HSTS as it is effectively enabled on all websites, and not just those where administrators choose to enable HSTS. The downside of this is that there may be some outlier false positives where an administrator chooses to disable HTTPS legitimately. In this scenario, a user of HProxy would be erroneously blocked from using the website. This scenario is probably unlikely; due to the low cost of maintaining HTTPS websites, there is no obvious reason why an administrator would choose to deliberately disable HTTPS. Much like HSTS, this technique would not be effective in Tor Browser as all history data is deleted upon closure.

Sivakorn completed a 2016 analysis [74] of websites which still only partially support HTTPS. Many websites at the time would mandate HTTPS connections for sensitive sections of their website, like login pages or purchasing. While this is a positive step, the authors show that an adversary performing a Man-in-the-middle attack can still obtain substantial personal information. In particular, the authors note the danger to Tor users who could be completely deanonymised by a malicious Tor exit node, even if the website supports HTTPS for the most sensitive pages.

Chung *et al.* [75] conducted a large experiment aiming to identify instances of connection sniffing or spoofing on the internet. The experiment uses the Hola unblocker which allows users to provide their own computers as proxies for other users. A user who is attempting to access blocked content can proxy their connec-

tion through one of the other user's computers in another country. This provides a wide set of 'Exit nodes' through which the experiment can request web content. By making web requests through different nodes, the authors were able to identify when and how content was changed from how it should seem and thus identified instances of man-in-the-middle attacks on users. Often, these attacks were being performed by ISPs and for the purpose of advertising. While not a Tor-based attack, this study highlights the prevalence of these attacks even outside the network where Internet service providers wield a similar power to Tor exit nodes.

Mendoza [76] found significant differences in the deployment of HSTS and HTTPS between mobile and desktop versions of websites. This indicates increased risk from all of these attacks to mobile users. Tor Browser is available on Android and Apple mobile devices, and a HTTPS-Only mode has also been deployed there.

Recently, Zhang *et al*. [77] proposed a set of innovative man-in-the-middle attacks against HTTPS connections. The attacks exploit the fact that different web servers can often use the same TLS certificate for their HTTPS connections despite having different domain names. Often, these are large corporations that use many different domains; Microsoft is a common example. The attacks allow a malicious intermediary (like a bad Tor exit node) to redirect secure HTTPS requests from one domain to another. Usually, the browser will accept the response from the second server as it has a valid certificate for the website which was requested. This can either (a) allow the attacker to chain re-directions to a website that does not require HTTPS or (b) redirect the user to a server which has poorer security configurations and use this to drop the HTTPS connection.

An important example of the type of security configuration that can be bypassed is HSTS. If the first server has well-configured HSTS flags, the attack cannot attempt to SSL strip; however, by redirecting the user to a server with a badly configured HSTS with low or zero duration, then this bad HSTS flag will be adopted by the browser for the first domain. The user can then be redirected back to this domain, which they originally requested, and SSL can now be stripped because the HSTS

flags have been damaged. The authors have contacted browser vendors to implement fixes; however, many of the attacks cannot be fixed by browser vendors alone and require specific fixes by the many web servers the authors were able to exploit. Adopting an HTTPS-only mode would mitigate all of the attacks mentioned. Interrupting and warning the user provides an opportunity to detect the attack and not proceed with an insecure connection.

Kampurakis [78] provides a survey of current web browsers as of January 2022 confirming that common SSL stripping attacks are still possible. The exception to the most basic form of SSL Strip is Firefox version 90+ in incognito mode, where HTTPS requests are upgraded by default. This only defeats the simplest form of SSL stripping attack. All browsers are still vulnerable to the more complex attack where HTTPS requests are downgraded to HTTP requests. Eliminating the simplest form of SSL strip will also occur imminently in Google Chrome as auto upgrades to HTTPS will occur [79].

Selvi proposed a new method for bypassing HSTS [80] by manipulating the computer clock and causing HSTS flags to expire erroneously. This is achieved by spoofing a Network Time Protocol response when the victim requests to update their local time. NTP is a UDP protocol; thus, no authentication or encryption is used. A man-in-the-middle attacker can easily spoof a response to such a request and thereby maliciously change a system's clock. This attack does just that and chooses a time far enough into the future so that all HSTS flags will have expired. Once HSTS flags expire, SSL Stripping attacks can be performed against the victim again. This attack is powerful; however, it cannot be performed by Tor exit nodes as UDP traffic is not routed over Tor.

Berta and De Los Santos produced a second HSTS attack [81], expanding on the technique used by Selvi. Firefox maintains a list of 1024 HSTS flags which is populated as the user browses to websites which provide HSTS headers. Once the list of 1024 is full, entries must be removed whenever a nwe HSTS flag is to be stored. In order to prioritise which entries can be removed from the list a score is main-

tained for ecah entry on the list. Every day the website of an entry is visited, the corresponding score of that entry is incremented. When a new entry is made, the lowest score flag is removed to make room for the new entry. The scoring system means that websites that are visited often are less likely to be removed from the list, and their security is maintained. The system also means that it would be difficult for an attacker to erroneously fill the list with junk HSTS entries, and thereby evicting legitimate entries from the list. Berta and De Los Santos use the system clock manipulation through the NTP protocol to bypass this safety feature. A man-in-the-middle attacker can, in quick succession, provide the victim's browser with junk HSTS entries and then manipulate the victim's system time to abuse NTP. The victim's clock is advanced 1 day in the future, and the attack again provides the browser with the same junk HSTS entries. This technique can be repeated as long as necessary for the victim's HSTS list to be destroyed. HSTS is then effectively temporarily disabled for the victim, who will be vulnerable to SSL Stripping attacks.

2.11.1 SSL Stripping in Tor

The Fort consulting company [82] presented a first deanonymisation attack that could be performed due to HTTP interception. A web visit via a malicious Tor exit node can insert an invisible iframe tag to the user's web request, which then requests a Shockwave object from another malicious server. That Shockwave object then launches a connection to the malicious server but bypasses Tor and reveals the user's real IP address. Unfortunately, the publication of this study also included real IP addresses and domain names for users who were deanonymised by Tor. This serious ethical violation significantly degrades the publication despite the real and noteworthy attack. Abbott [20] expanded on this attack and presented a traffic analysis attack which used the same basic principle of injecting iframes into HTTP requests. The attack avoids using Shockwave, which is now blocked in Tor Browser and implements an HTML-only attack instead. The malicious iframe continuously sends signals back to the malicious server via the Tor circuit. If the users leave this malicious tab open, over time, the Tor entry being used would change to a malicious one, and traffic could be correlated between the malicious node and the

malicious server. This attack, however, is no longer possible as Tor guard nodes are used for all entries to the network, and those guards rotate very infrequently, on the order of months. While both these attacks are no longer possible in the modern Tor Browser, they still illustrate a system where a lack of HTTP connections can cause serious damage and the potential for new deanonymising attacks is always enabled by malicious exit nodes having the ability to manipulate users' HTTP traffic.

2.12 Other HTTPS Indicators

While the methods for automatically upgrading and automatically enforcing HTTPS have been useful, there do not fully protect against SSL Stripping attacks. Aside from these automatic methods, there have also been user interface changes that have attempted to show users where they are or are not using HTTPS. The intention behind these indicators is to allow users themselves to make an decision based on the security of their connection. Ina situation where automatically upgrading to HTTPS has failed silently, and HSTS is not in use, it can fall to the user to notice that they are not using a HTTPS connection. This allows the user to decide whether the website they are visiting should require a secure connection and potentially to navigate away from the website if they consider themselves at risk.

These types of indicators have existed for a long time in most popular web browsers. Most commonly, this is found in the usage of the 'padlock' symbol, a completely ubiquitous indicator that can be found in all browsers, usually beside the URL bar. Its presence indicates that the current web page is being connected to through HTTPS. Much work has also been done on evaluating the effectiveness of these HTTPS indicators, mainly URL bar security symbols.

Felt *et al.* evaluated them in 2016 [83] and proposed new indicators to convey information to users properly. Kraus [84] outlined how these indicators have evolved over the years for all major web browsers. In particular, Firefox and Google Chrome's move to using 'insecure' notices for non-HTTPS websites is discussed.

Thompson *et al.* [85] also found that the special 'EV' security indicators had no

effect on users' behaviour.

Dhamija *et al.* [86] completed a small qualitative study and found that participants did not use browser security indicators (i.e. 'the padlock') to help them make informed security decisions. Many participants specifically stated they did not know what the lock was, or they did not consult it as an indicator of how secure the action they were taking was. Despite the age of this study, the authors make a perennial conclusion: that security indicators should appear not just when things are safe, but there must be a specific indicator to users when things are not safe.

Whalen [87], in a limited study, found that users did look at the 'padlock' browser security indicator. It was not clear whether users fully understood the indications of this or if they used it to alter their security assessments.

Schechter *et al.* [88] showed that URL HTTPS indicators were not particularly effective at informing users, and when the indicator was removed, users had the same tendency to visit sensitive websites and enter sensitive data.

For most of their life, URL bar indicators have served as a confirmation that HTTPS was in use. There was not, however, a contrary indicator showing users when it was not in use. This eventually changed, and now all major browsers provide an insecurity indicator to indicate that the current connection is not secure.

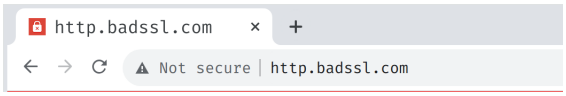


Figure 2.6: The URL bar insecurity indicator in Google Chrome which is displayed whenever a non-HTTPS website is visited.

Browser warnings have also previously been used to warn users against 'mixed content' web pages where a HTTPS connection also loads external non-HTTPS resources to populate the webpage. Internet Explorer previously included a notorious mixed content warning which was likely not intelligible or useful to users. This can particularly be seen through the numerous articles informing users how to disable the warning messages that are available online. Jackson *et al.* [62] argued that

warnings like this were mostly ignored by users, primarily because of their very high prevalence in daily browsing. As an example, for many years, the Gmail home page contained mixed content and thus caused such a warning to display. Such a common error appearing on a popular website likely habituated users to ignore the warning and thus made it ineffective.

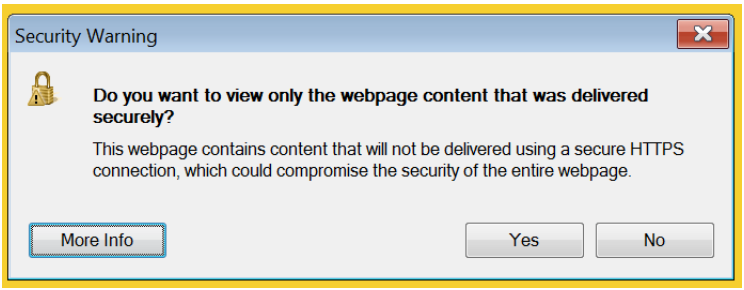


Figure 2.7: The mixed content warning previously displayed in Internet Explorer. It is clear that the warning provides no guidance for users who do not understand what HTTPS is.

It is clear that Microsoft did attempt to improve users' response to the warning [89]; however, it is not clear that there was any effort to improve user understanding of why the warning was appearing in the first place.

In 2011, Google announced that insecure script resources would be blocked by default on HTTPS webpages [90]. In 2015, Google Chrome changed the design of their URL bar padlock indicator [91]. Websites with mixed content would no longer benefit from a padlock with an error icon. Instead, mixed-content webpages receive no lock icon, just like a webpage that does not implement HTTPS at all. In 2016, Google announced that non-HTTPS webpages that collect input data like credit card numbers and passwords would be marked as not secure [92]. In 2017, Google announced that all non-HTTPS webpages in Incognito mode would be marked as insecure [93]. In 2018, Google announced that all non-HTTPS webpages would be marked as insecure [94]. In 2020, Google announced that a warning would be added to forms that were about to transmit over non-HTTPS connections [95].

Google has also begun experimenting with removing the lock icon altogether due to users often misinterpreting the purpose of it [96].

Shin presented SSLight [97], which is a system for inducing traffic light-style warnings onto forms in web browsers. The browser extension software runs a simple algorithm to determine the website's security. Suppose the form action points to the same website that is currently loaded. In that case, if the connection will be made via HTTPS and if the SSL certificate being offered is valid, the form will be affixed with a green light to indicate security. If the form points to a non-HTTPS site or if the certificate is invalid, a Red light appears on the form. If the certificate is valid but does not point to the same website, a whitelist is checked, and presence on the whitelist awards a green light. If the website is not on the whitelist, an Orange light appears which is intended to advise caution. The authors also include a technique which flashes the form background in the chosen colour instead of a simple light at the end of the form. The authors perform small usability studies into the effectiveness of these methods and find that there is no benefit from using the traditional popup style warning that was used in Firefox at the time compared with no warning at all. The authors find significant benefits from using their new SSLight scheme and the flashing background method, with the flashing background method having an even higher rate of stopping users from submitting personal data to insecure forms.

Having critically analysed this work, although it seems the new method for warning users of insecure forms is effective, the benefit gained is potentially entirely due to a lack of user habituation to the new warning methods. Many of the works I have covered identify habituation as a significant factor for why security warnings fail to make users take secure actions [98, 99, 100]. Additionally, studies which specifically design new methods for avoiding habituation [101, 102] seem to present with success. It is, therefore, likely that the SSLight scheme succeeds experimentally mainly because of its novelty, and as such, implementing a system like this would soon fall prey to habituation, just as other schemes have done. That being said, compared to the popup insecure form warnings which were in use at the time, SSLight does have merit in reducing the complexity of the warning. Removing all text from the warning and instead focusing on visual indicators could potentially have pro-

vided benefit by reducing the level of effort required by users to understand what the warning is attempting to convey. Nevertheless, as was highlighted by Krol's [98] reducing false positives should be the primary way of reducing habituation to warnings like this.

2.13 HTTPS-Only Modes

HTTPS-Only modes are a relatively new option in most popular web browsers that assists in forcing HTTPS connections to websites. When enabled, this option first automatically attempts to load all web connections via HTTPS. Secondly, if the website does not support HTTPS, a warning is displayed to the user, and they can choose whether they wish to continue to the website without the protection of HTTPS. This, of course, is a useful security feature for maintaining security while browsing the web. HTTPS-Only mode was first introduced in Firefox in 2020 [103], in Google Chrome in 2021 [104] and in Microsoft Edge in 2021 [105]. Safari does not include a mode that warns users about proceeding to HTTP pages; however, all requests are now automatically upgraded to HTTPS if it has been available since Safari 15 [106]. The Brave browser also includes a HTTPS-Only mode, it is the only browser aside from Tor Browser to enable the mode by default [107]. All other HTTPS-Only modes are optional non-default settings.

Automatically upgrading requests to HTTPS provides enhanced security at no cost or disadvantage to users. Displaying a warning to users can also help them recognise sites where it may be dangerous to proceed to a non-HTTPS website. These warning modes are a natural progression from previous browser indicators. The URL bar indicators that were discussed served to notify users that an insecure HTTP connection was being used. As HTTPS adoption has increased in recent years, it is logical that warnings to users can become more intrusive as they become less frequent. It is clear that the adoption rate of HTTPS has increased dramatically over the past number of years. Websites with HTTPS as default rose from 26.9% in 2018 to 84.9% in 2024 [108]. It is also the case that smaller, less intrusive indicators to users are not as effective at convincing users not to proceed to websites [88].

The transition from URL bar indicators to full page warnings provided by HTTPS-Only modes provides users a much more explicit warning. It can alert them to danger in both cases where they may be visiting a new site that does not have HTTPS enabled or, perhaps more seriously, alerting them that a site they usually visit and normally does have HTTPS.

The previously discussed addition of non-HTTPS URL bar warning indicators in both Chrome and Firefox over past years is a development that both enables and assists HTTPS-Only modes. For many years, users have been exposed to these warnings indicating that HTTP websites are 'not secure'. While this does not provide a nuanced view of where it may be appropriate to continue to view legitimate non-HTTPS websites, it does provide a growing distinction for users between secure HTTPS and insecure HTTP. This may assist to some degree in countering findings by Krombholz [109] that users often do not consider HTTPS a necessary or useful addition. These changes, having taken place several years before the arrival of HTTPS-Only modes, have primed users for their arrival in this way. However, They will work in tandem with these modes by reminding users if they have proceeded past a warning page to an HTTP site and encouraging them to keep in mind the risks outlined by the warning page. This wider trend of increasing scrutiny of non-HTTPS websites results from increasing HTTPS adoption in recent years. HTTPS-Only modes form part of this trend. As the adoption of HTTPS continues to rise, the potential for harsher measures warning against non-HTTPS websites appears more likely. This follows from Krol's [98] work on false positive habituation which is discussed in Chapter 5. As warnings are seen by users less often, they can be more disruptive. They can demand more cognitive effort from users without contributing to the habituation of these and other warnings.

In practice, HTTPS-Only modes typically operate by first altering all requests made by the browser from HTTP to HTTPS. Any requests not receiving a response within a timeout period cause a secondary background request to that website on a downgraded HTTP connection. If that connection is successful, it indicates that the web-

site does exit, but does not support HTTPS. A warning page is then displayed to the user, typically with the option to continue to the insecure version of the website or to close the website.

An analysis of the current HTTPS-Only mode Warning pages offered by major browsers is provided in Chapter 4, Section 4.4.

2.13.1 Tor Browser and HTTPS-Only Modes

Tor Browser has included the HTTPS Everywhere addon since 2010 [57]. This has meant that websites are automatically upgraded to HTTPS when available. HTTPS Everywhere also featured the 'EASE Mode' function, which acted identically to an HTTPS-Only mode; however, the Tor Project has never recommended enabling this option as selectively enabling it would make the users' browser fingerprint different to other Tor Browser users [54].

Reducing the danger that is faced from non-HTTPS websites has long been a priority of the Tor Project. Discussions have taken place since 2016 into how this would be achieved, however, adoption of a default non-HTTPS blocking mode did not occur until 2022 [110]. Tor Browser is ultimately an updated packaging of Firefox, and thus, the HTTPS-Only mode added to Firefox was imported into Tor Browser in 2022. It was also enabled by default in release 11.5 [111] fulfilling this long ambition to reduce the risk users faced from non-HTTPS connections, in particular due to the danger posed by bad exit nodes in the Tor network.

2.14 HTTPS Adoption

The effectiveness of HTTPS-Only modes could largely depend on the level of adoption of HTTPS on the Internet. If HTTPS adoption was low, users would frequently be faced with false positive warnings that interrupt their browsing experience. Conversely, a high rate of HTTPS adoption ensures that users see false positive warnings infrequently, and the presentation of a warning has a higher chance of indicating that an attack is taking place.

HTTPS has become more widespread, after growing steadily for approximately the last decade. The release of the Firesheep [112] Firefox addon in October 2010 perhaps heralded the start of a new era where HTTPS was no longer considered optional. The addon allowed users to easily obtain the session authentication cookies for other users on their network. This worked for a variety of different websites, but most notably Facebook. The ability to easily obtain access to other users Facebook accounts likely underscored the necessity of using HTTPS connections for all website where a user was logged in. HTTPS adoption by websites is often seen as a simple binary characteristic. In the case of the Firesheep addon and Facebook, only the login page of Facebook provided HTTPS and connections to the other pages of the website were vulnerable to eavesdropping.

Currently, most popular websites that deploy HTTPS do so for their whole website. However, Paracha *et al*. [113] showed that this is still sometimes not the case. Their study surveyed a total of 110,000 of the most popular Alexa websites and found that 1.5% of did not have HTTPS available for all parts of their website. Additionally, 3.7% of the websites returned different content over HTTPS and HTTP for some sections of the websites. These proportions are not particularly substantial, but they are large enough to remind us that HTTPS adoption is not always a binary criteria.

Similarly, when measuring HTTPS adoption, some measurements discuss HTTPS availability meaning that the website does support HTTPS, while other measures discuss default HTTPS adoption. Default HTTPS adoption denotes a website that automatically redirects its visitors to HTTPS connections and usually does not allow non-HTTPS connections.

As a proxy for HTTPS availability, HTTPS adoption currently is measured at about 84.9% by W3Techs [114] for default protocol HTTPS and Google has measured 90% of navigations using the Google Chrome browser as being to HTTPS domains [115]. This means that while a large proportion of website visits are likely to be HTTPS, there are still a reasonable number of visits to HTTP sites under legitimate circumstances where the website has chosen not to implement HTTPS.

Felt *et al*. [116] attempted to measure HTTPS across the web in 2017. This proved a difficult challenge due to the many possible interpretations of HTTPS adoption. Nevertheless, the result showed a continuing increase in HTTPS adoption, which has continued in the years since the release of this paper.

2.14.1 Lets Encrypt

Increasing adoption has potentially been mainly facilitated by the launch and growth of LetsEncrypt [117, 118], which allowed web administrators to obtain an SSL certificate free of charge. It is clear that LetsEncrypt has dramatically transformed the state of HTTPS adoption on the web, providing a certificate to 35% of the top one million sites on the web as of January 2019 [119]. There is also evidence that obtaining a LetsEncrypt certificate is easier than traditional methods, beyond the benefits of free pricing.

Bernhard *et al*. [120] conducted a study comparing the LetsEncrypt process to traditional processes and found that LetsEncrypt had simplified several aspects of the process; the study concludes that LetsEncrypt is a slightly easier process to follow.

In 2017, Hunt [121] argued that HTTPS adoption had reached a 'tipping point' where HTTPS would become the default option for web administrators. He argues that this is largely fueled by LetsEncrypt providing free certificates but also by Browser vendors implementing features that discourage HTTP use.

in 2018, Mirian *et al*. [122] conducted a study of HTTPS adoption for 'long-tail' websites, this survey studied the top 1 million websites, but excluding the top 10,000 websites for which HTTPS adoption is extremely high. In these remaining sites, adoption was much lower at 45%. The authors found that in these long tail sites, use of LetsEncrypt was 4 times more likely than in the top 10,000 and that websites which switched certificate authorities were 3.7 times more likely to switch to LetsEncrypt from another provider than in the top 10,000.

While the authors were not able to find any other significant findings to assist

HTTPS adoption in this long tail, they did find that easy provision of HTTPS through cloud hosting providers was a significant factor in increasing HTTPS adoption of these websites.

The level of HTTPS adoption by websites is an important consideration for HTTPS-Only modes. The higher the adoption rate, the less often users will be faced with warnings. If all websites supported HTTPS, users would only be faced with HTTPS-Only mode warnings in risky situations, and as such, the warning pages could reflect that by designing a fearsome and explicit warning that encouraged users not to proceed to the non-HTTPS website.

Chapter 3

Anonymous Collaboration: How Does Delay Affect User Experience?

3.1 Introduction

The mixnet is one of the most powerful anonymity systems available to Internet users. Although mixnets can provide very strong levels of anonymity, they achieve this by adding high levels of latency to communications. Previous iterations of mixnet design like Mixminion [123] and Mixmaster [9] worked with very high latency on the order of hours [124]. As a result of this latency, mixnets have historically only ever been used for non-real-time communication, mostly emails [125]. Recent developments in mixnet design have allowed their use with lower latency and, as such, potentially allow the use of real-time protocols. Most notably, the Loopix mixnet system [13] incorporates many new techniques that will enable much lower latency to be deployed whilst still maintaining high anonymity guarantees. Deploying this protocol with its lower delay could allow a wider variety of applications to be used on mixnet systems. To test this hypothesis, I developed and ran a user study to evaluate users' tolerance to delay as would be caused by modern mixnet systems. Given that the types of activity performed online can vary drastically, and thus, the impact that delay has on users will vary with this, I decided to attempt to produce a particularly difficult scenario for examination. This would

mean that the result of the study can be considered a relatively high bound on the difficulty induced by that delay. I deemed this difficult online scenario to be two users collaboratively completing a real-time typing task. By testing users' responses to this typing task, I developed critical insights into users' behaviour when faced with high latency. As a result, I provide key recommendations to modern mixnet operators on finding the appropriate latency levels to use. I also offer the first real context for mixnet delays and show future mixnet researchers a key metric for the usability of their systems.

The core of the study involves producing a web application that simulates, to some degree, the environment a user would experience when using an actual collaborative application that has increased latency from a mixnet. This consisted of a series of general knowledge questions which participants answered in collaboration with a simulated second user. Communication between the users is operated under similar delay characteristics to the Loopix mixnet. That is, the delays faced for messages in the Loopix mixnet are sampled from a Poisson distribution, so messages in this study are also delayed using a Poisson distribution. This ensures that this study can be used as a parallel for how users of the Loopix mixnet would tolerate delay.

Naturally, a user's tolerance for latency in an application is a qualitative data. As such, the results of this study are not a binary "tolerable" or "not tolerable" but a relative score based on qualitative questions and other data. My study measures only a select few latency values and does not provide a full distribution of users response's to all different possible latencies. The latency values selected for study provide a useful look into the general progression of user response as latency rises.

3.2 Research Questions

I utilise the following research questions to inform how the study was designed to asses user's responses to this latency.

- At what level of delay does collaboration slow beyond usefulness?

- What are the specific reasons for collaboration breaking down at that level?
- How does user frustration vary as delay increases?
- Do users tolerate an increased delay, even after it renders the whole task slower?
- Does increased delay affect task completion rates?

3.3 Structure of the Chapter

This chapter first discusses some background information in Section 3.4; it then outlines some related studies on user delay, discussion of different collaboration architectures, collaboration data structures, and previous levels of delay for mixnet systems. Section 3.6 will provide an overview of the study itself, detailing the creation of a simulated user for collaboration with study participants and other design considerations. Section 3.7 shows the study results, and finally, discussion is provided in Section 3.8.

3.4 Related Work

The user study I designed on the usability of anonymous applications is novel. I consulted other related studies in the literature which are outlined here with notes on their commonalities and points of interest, which have assisted in the design of the study.

3.4.1 Studies on Text Editing

Ignat *et al.* [126] presented an important example of a user study focused on collaborative applications. The study aimed to evaluate how the quality of users' prose is affected by delays introduced in a collaborative text editing environment. The task chosen by the authors is influenced by conclusions from Olsen and Olsen [127]. Olsen and Olsen claim that the closer two collaborators tasks are to each other the more problems there are in collaboration. Tasks with close interaction between participants are said to have high coupling. Ignat thus decides that a medium coupling task would provide a reasonable measure and chooses collaborative note-taking as

the study task. In collaborative note-taking, participants edit text that is reasonably near to each other, but not directly beside thereby making it a medium difficulty collaboration task as regards conflict.

This study evaluates the effect of delay caused by everyday web browsing, congested internet connection, or delay from the collaborative data structure itself. Therefore, the level of delay being expected is not likely to be on the same level as in the study owing to the larger delays needed for anonymity systems. That being said, some key observations were made in this study. The authors note that any delay introduced which is less than the participants' average typing speed is unlikely to have any effect on satisfaction levels and so can be discounted. To ensure the study's effectiveness, a strict outcome metric is necessary. This metric will clarify when users succeed and when they do not, directly influencing the experiment's results, making it crucial to choose it carefully. As the authors are dealing with prose written by participants, they endeavoured to create a metric on the quality of the prose itself. The transcription task carried out by users was susceptible to grammar and spelling errors, as well as duplication errors by the multiple participants.

The authors' research questions revolved exclusively around how the delay added affected quality measures on this transcription. The authors conduct a preliminary design simulation to find what levels of delay occur between users in a typical Google Docs session. This establishes a baseline of what delay level users face on high-quality internet connections and through the efficient collaboration data structure of Google Docs. This simulation produces a graph of the resulting delay in receiving messages based on the number of group members from 1-38. Large delays occur when large numbers of users are editing the document. For example, with 30 users, delays of 6 seconds can be seen between when a letter is typed and when it appears on another user's screen. This number of users is not relevant, as only smaller scenarios are considered in my work, typically two users. The authors do find that for smaller groups of less than five persons, a delay of 1 to 2 seconds can occur. From this data, the authors decided on using 4,6,8 and 10 seconds as the

different induced delay options in their study. Ultimately, the authors find a strong correlation between user errors and the level of latency introduced.

Kopsell performed an experiment into tolerance for delay in anonymity systems [128]. The experiment was performed on the JAP anonymity network which was a mixnet style anonymity network. At the time of experimentation, most of the traffic for this network travelled through servers operated by the authors. The authors were able to periodically add additional artificial delay to their servers and monitor the reaction from users. The results, over many months clearly show a linear relationship between the number of seconds of additional delay and the decrease in users of the service.

Finally, Dang and Ignat conducted a study of online document edits at scale [129]. Although the study is primarily aimed at testing collaborative editing with large numbers of users, it reminds us of the very low sub-1 second delay present with Google Docs and the open-source EtherPad application.

3.4.2 Delay's Impact in Other Scenarios

Bai *et al.* [130] studied users' latency tolerance in web search. The study finds that less than 500ms of delay is not reliably detectable by users, and greater than 1000ms of delay is reliably detectable by users. Interestingly, the authors prepare a machine learning framework for detecting how susceptible users are to delay. This potentially shows that different users perceive and experience delay differently although more study is likely needed to confirm this finding generally.

Arapakis *et al.* [131] completed a similar study about web search. The authors conducted an experiment to test participants' responses under different levels of delay. Participants were given one of a set of complex research tasks that involve investigating a particular topic and finding a list of items in that area. The tasks were designed to require a number of different search queries. Different levels of delay were added to the search engine for each task. They employed a super-linear transformation $l(x) = 1.5^x * 100$ to compute the different delay levels tested. This

was done as, generally, 'human perception is not linear.'. After the search task was completed, participants were given several Likert scale[1] questions to evaluate their experience. A Friedman test was used to identify general significant difference across all questions. A Games Howell post-hoc test was used to find pairwise differences in each Likert scale for all latencies. The specific study task does not align with my study aims; however, the general task design mirrors my own. The authors found that mobile users are more tolerant than desktop users to latency, but that nevertheless, after approximately 7-10 seconds of delay, users show signs of negative feelings and an overall less satisfactory experience.

Gergel *et al*. [132] conducted a study on how visual delay affects users. The study task consists of a visual puzzle. One participant acts as the puzzle solver and can take action. The second participant can see the finished puzzle but cannot take action. The participants must collaborate using a voice connection to solve the puzzle. Delay is added to the second participant's view of the puzzle, so they will see a delayed view. Generally, the delay level had less of an effect than in other scenarios found in the literature. The authors admit this themselves, and design a more difficult task with dynamic changing puzzle pieces. This change brings the results more in line with previous work.

Nah *et al*. discuss the time users are willing to wait for a web page to load [133]. The study involved approximately 40 participants who took part in various web page loading simulations. The study evaluates users' tolerance for loading speed using an exact metric: how long a user waits until they give up and close the web page. There is an obvious cutoff point where it is clear a user's tolerance has reached its limit. This contrasts with the study goals, where users' frustration is not as readily apparent. While the results of this particular study may not be directly relevant, it still provides valuable inspiration for my similar user study.

[1]A Likert scale is a simple rating scale used to assess user opinions. It typically consists of 5 levels users can choose from. Users can provide an answer between 1 and 5 to signify how much they agree with a statement. Typically with 0 being totally disagree and 5 being strongly disagree. Usually 3 is considered the neutral value.

Galletta *et al.* [134] also conducted a study on tolerance of website delays. Generally, the study confirms previous findings that there is a significant effect on users from increased delay, but also the study showed that a wide variety of factors can affect this mechanism, including the type of website task as well as users familiarity with that task.

Van Damme *et al.* [135] studied the impact of latency up to 500ms on a virtual reality collaborative task. Participants reported more difficulty with higher levels of latency. Participants were faced with different types of latency, primarily using differences in the burst levels of the latency. Participants had difficulty distinguishing between high and low levels of constant delay with high bursts of latency.

3.4.3 Effect of Delay is Task Dependent

Beigbeder [136] showed how increased latency affected players of the Unreal Tournament First Person Shooter game. They concluded that different interactions with the game were susceptible to increased latency in different amounts. Moving around the game world, for example, was a task that was not effected significantly by increased delay or by significant packet loss. Aiming and shooting, however, were dramatically effected, with even delays of 100ms decreasing users' accuracy by 50%.

3.4.4 Users Come up with Strategies

Vaghi *et al.* [137] experiment with user tolerance of delay in a simple networked game. Two study participants controlled the simple football-style ball game. Different levels of delay were added for each pair of participants, and the result was noted. They found that delays of up to 150 milliseconds were not noticeable to participants. From 150-300 milliseconds, the participants noticed some errors, for example, users attempting to kick the ball as it appeared on their screen. Still, due to divergence between the user session and the central server, the actual ball location was not where the user expected it, and the kick failed. This was noticeable but did not entirely ruin the game experience. At 500 milliseconds, the game was increasingly frustrating and challenging to play. At 999 milliseconds, playing the

game was impossible anymore; the ball could almost never be hit.

After interviewing the participants, the authors found the players were employing a series of interesting coping strategies or techniques. Some players attempted to slow the ball down to reduce the delay's relative effect on its whereabouts. Some players used more long-term prediction of the ball's trajectory and waited for a longer period at a point where the ball would eventually arrive or ensured that the user's avatar was always placed somewhere along the trajectory of the ball so that a collision would occur even if the player does not currently know the actual position of the ball. This highlights an important finding in users' tolerance of delay which is that they can adapt to specific situations to improve their experience. Users finding different strategies to avoid the consequences of delay is likely to generalise to other forms of collaboration. The exact strategies in the authors' study, of course, do not apply to other more common tasks like collaborative editing. Nevertheless, users are expected to learn how to adapt to any scenario with delay.

3.4.5 Application Design Choices Mitigate Delay

Various techniques have been used in online collaborative applications to alleviate the effects of latency. Often, these are psychological tricks to make users more tolerant of latency issues. Savery *et al.* [138] highlights the example of the online game 'World of Warcraft', where play characters under heavy latency would ordinarily be shown to 'jump around', proving jarring to users. Instead, the local software guesses a reasonable path for the character to move from point A to point B, and the character is shown to gracefully walk the route, thereby avoiding jarring teleportation and enhancing the user experience under heavy latency. This is a relatively simple example of a broad range of techniques that can be employed in all collaborative applications.

Gutwin [139] proposed using 'Traces' to enhance the usability of collaborative applications. The previous actions of collaborators can be displayed to users in the hope that users can more effectively predict where the user will act next, thereby neutralising some of the negative latency effects. This trace can be displayed in

various ways depending on the application. One example given by Gutwin is a user's cursor. A trace, or shadow, of where the user's cursor has previously been is displayed, allowing them to predict the other user's intentions to some extent and predict where they will move their cursor next. The authors also investigated whether informing users of the level of delay currently being experienced allows them to predict the actions of their counterparts better. The authors specifically targeted jitter and test users' abilities with different levels of jitter, not latency. The techniques employed show significant results; users were better at predicting where a user's cursor would end up if the decorator was added, telling them the delay at each update point.

In another study by Gutwin *et al.* [140], the authors developed a system for handling disconnection in collaborative systems. Disconnection can often be due to delay in the system. The authors' work showed that the application interface can be altered to assist users' experience considerably even in the face of issues.

Yeh *et al.* [141] studied two different methods of improving user experience with collaborative document editing by delaying the display of user's edits to other users. Several interventions are studied, for example, delaying characters from being displayed until they form words or delaying sentences from being displayed until they are completed. Participants reported their feelings across a number of different metrics, including satisfaction and ease of use. Overall, the authors found that many of these interventions can be effective in different ways. This further shows the ability to customise user interfaces to increase satisfaction or mitigate issues that can arise.

Khalid *et al.* [142] completed a study investigating mitigations for delay in a collaborative virtual environment. As with the other previously mentioned works, this work implemented various user interface tweaks to make the user experience more comfortable and mitigate the issues caused by latency in the application. The authors used the metric of task completion time to assess users' ability to use the application effectively. Delay was then added from 50ms to 200ms. The visual pointers and guides managed to mitigate some of the reduced completion time seen

from the increasing delay. This study has a similar design to my own, which will be discussed. However, it deals with a 3d virtual reality task and considers much lower added latency levels.

3.5 Background

3.5.1 Typing Speed

Dhakal [143] conducted an extensive study of typing performance characteristics on 168,000 volunteers. One key result is that participants' average typing speed is 51.56 words per minute (WPM). This gives us a baseline figure from which to measure the typing performance of study participants. This is equivalent to 0.86 words per second. Estimating the delay per character requires the average length of words in the English language. This is estimated at 4.7 [144]. From this data it can be computed that the average user likely types at approximately 4 characters per second; thus, the delay between characters being typed should be 250ms. A constant factor delay would not perfectly simulate a user's typing patterns, so the simulated user samples a delay from a Poisson distribution to add unpredictability to the pauses.

3.5.2 Collaboration Architectures

Achieving consistency in collaborative documents is usually done using one central data authority. Typical collaborative solutions like *Google Docs* are conducted over a central server, and any edits done are reconciled directly and immediately with the authoritative document on the server. In many cases, this makes for a more simplistic and straightforward protocol than a peer-to-peer model in which the service users communicate directly with each other. The success of services like Google Docs shows the viability of centralised services with algorithms like Operational Transform; however, it also poses some additional problems for introducing anonymity. In a centralised model, the server necessarily requires access to the data being operated on. Even if the participants cannot be identified based on IP address, the data in the file will likely compromise their anonymity completely. Additionally, the need for packets to traverse the central server doubles the latency,

and given that latency is one of the most difficult problems in anonymity systems, it seems crucial that a peer-to-peer system is used for the task.

Centralised models are the most popular choice regarding real-world usage and appearance in the literature; however, peer-to-peer models also exist. In Van Hardenberg's PushPin [145] and Nicolas' MUTE [146], some of the challenges of building a fully functional p2p application are outlined, and there are some difficulties introduced in a p2p model, like NAT traversal. Overall, it is clearly shown that peer-to-peer collaborative settings can properly work in a real-world deployment. Kleppmann *et al.* [147] discuss the advantages of peer-to-peer collaborative applications that store all user data locally, the article both espouses the benefits of these applications but also their practicality due to new technologies such as Conflict-Free Replicated Data Types (CRDTs).

Peer-to-peer communication models cannot use a central server as an arbiter of document truth. Thus, popular collaboration algorithms like Operational Transform cannot be used. Modern data structures like the CRDT must provide a robust peer-to-peer collaborative algorithm with accurate results.

3.5.3 CRDT's

A Conflict-Free Replicated Data Type or CRDT [148] is a concept introduced to address some of the issues of Operational Transform. Fundamentally, while Operational Transform works by altering or transforming collaborative operations such that they do not conflict before they are applied to the data, CRDTs take a different approach. The approach of CRDTs can be quickly summarised as constructing a data structure whose operations are commutative from the beginning. Thus, parallel instances of the data across a network will always converge to a single true state, even if the operations are carried out of order on different instances. A wide number of CRDTs have been proposed in the literature. Of course, they differ in the data being stored and the types of operations that can be conducted. However, the most sought-after CRDT is typically the one that allows collaborative text editing.

One of the earliest CRDTs, *WOOT* [149], allows editing large arrays of characters with both insertion and deletion operations, but it also grows as a data structure without bound, as deleted characters are stored as 'tombstones'. The latter, *treedoc* [150], implements a garbage collection protocol which prevents the structure from becoming too big but still results in an overall storage size increase. More modern CRDT protocols still suffer somewhat from the issue of storage but considerable improvements have been made. Nédelec [151], Weiss [152], and Kleppmann [153] have all made advancements in this space. However, the general traits of CRDTs remain the same. Compared to Operational Transform systems, CRDTs allow for true peer-to-peer collaboration without needing a central server. Still, this is at the cost of increased metadata storage overhead and overall protocol complexity.

For my interests, there is no other choice than using a CRDT: not only is the lower latency and fault tolerance brought about by the peer-to-peer architecture extremely desirable for anonymity applications, but the lack of a central server coordinating users has tremendous advantages for anonymity and privacy. In general, the potential for a central server to be corrupted by an adversary or, more generally, targeted for attack makes it an undesirable setup for anonymity purposes; additionally, an end-to-end encrypted collaborative protocol is likely incompatible with most protocols involving a central server. In other protocols like instant messaging, the data being encrypted is not an issue, as often the only role of the server is to forward the data to the receiver. In collaborative editing however, it is often the case that the central server must apply edits and resolve conflict in the protocol. Using end-to-end encryption with a non peer-to-peer model is therefore likely difficult. Therefore, the CRDT is the preferred choice. When pursuing anonymous communication, both latency and bandwidth are paramount, and having a robust and efficient CRDT or other data structure behind a system will help reduce the latency and number of messages in a collaborative protocol whilst allowing data to be kept private.

Ahmed-Nacer *et al.* [154] performed experiments into the performance of CRDTs versus more typical operational transform algorithms. The authors obtained traces

of large sets of collaborative document editing and re ran these edits through a number of CRDT algorithms as well as one traditional operational transform algorithm. The experiments clearly show that the CRDTs performance is on the order of the operational transform algorithm, and in some cases, certain CRDTs algorithms exceed the performance of operational transform. These experiments serve as a key indicator that collaborative editing can be done using CRDTs without a sacrifice in performance. Moreover, this indicates that CRDTs are the ideal choice for collaboration over high-latency anonymity networks.

3.5.4 Automerge

Automerge [153] is a project attempting to produce consistency algorithms without a central authority server . Essentially, it is a CRDT specifically designed for applications with poor network connectivity amongst other factors. In particular, it can run only on client-side operations on data with no central server or authoritative figure in the protocol. It is for these reasons that this protocol is particularly interesting. Peer-to-peer style protocols can send messages directly from user to user and thus no time is wasted sending messages through a 3rd party server. This is imperative for running the protocol over an anonymous network; in addition, CRDT's purported robustness in network drops, etc., is deeply appealing when paired with the sometimes unpredictable and strange behaviour of packets travelling through anonymity networks and mixnets in particular. Of particular note is the character level operations in the automerge protocol. It is tolerant of different users performing edits on different characters even within the same word of a document. This brings to light another trade-off which can be made to enhance performance. Making assumptions based on user actions can benefit us tremendously when designing protocols. Suppose, for example, that all user edits will not edit the same word in quick succession. In that case, the granularity can be decreased for the chosen CRDT or other data structures, which gives us simplified data structures.

Automerge not only guarantees the eventual consistency of collaborators but also comes with good performance. I selected Automerge for the study due to its low

latency and minimal overhead. This enables measuring only the effects from the artificial delay introduced and not also additional delay from the CRDT operations. An accurate picture of participants' response to delay should be produced, showing the effects, in the worst-case scenario, of the best possible technology. Results from an older, less efficient data structure would also not generalise.

While the Automerge protocol and library are useful for the study, it is also true that real-time collaborative applications with strong anonymity guarantees would potentially never be possible without modern CRDTs like Automerge and yjs [155]. The poor performance of older CRDTs resulted in very high latency operations or loss of critical features like deletion. More recent analyses show that CRDTs can support collaborative document editing without too many constraints [156]. Jahns also provides a comprehensive benchmark of the Automerge and Yjs CRDTs showing most operations requiring on the order of milliseconds to complete.

3.5.5 Appropriate Mixnet Delays

My study aims to measure user tolerance to the delays introduced by mixnet systems. To decide what level of delays should be considered, it is important to first discuss the history of mixnet delays.

Serjantov and Murdoch [124] conducted some evaluations of the deployed Mixmaster and Mixminion systems in 2006 and obtained statistics on the delay experienced in the systems. Mixmaster's median delay was 2.7 hours, and the minimum latency was approximately 13 minutes. These figures are only a brief snapshot of the current state of the systems at that time, and the delay characteristics can change rapidly depending on the number of users of the system as well as various other factors. Regardless, this is a strong indication that these early mixnet systems would not be able to support any real-time communication or collaboration systems, as the delay required is too high to make these applications usable.

There has only been a limited number of modern mixnet designs which show the improvements made in recent years. In particular, many messaging systems have

been designed that evidence the potential applicability of mixnet systems to real-time communications. These modern systems either use artificial delay added to packets/messages to enhance anonymity or use artificial noise or fake traffic. The net result of both techniques is increased end-to-end latency for the user's data. Vuvuzela [157] is a system for anonymous messaging that provides differential privacy. The system manages to provide messaging with, on average, 37 seconds of latency.

Stadium [158] is a real-time messaging system based on a mixnet design which provides anonymity with a delay which, in the best case, is on the order of 50 seconds. The Karaoke [159] system is a similar design but with improvement to reach delays of approximately 6.8 seconds. Karaoke also provides a much more robust evaluation of its anonymity guarantees and, at this rate of latency, can provide a differential privacy guarantee against deanonymisation for what the authors estimate to be one year of long-term data leakage.

Atom is another anonymous system [160] which permits microblogging. It offers high scalability and anonymity guarantees. However, much like older mixnet systems, it offers much higher latency communications, which can be on the order of several minutes in the best case. Systems like this demonstrate the further usefulness of mixnets in modern systems but underscore the more typical usage of mixnets as high latency systems.

Groove [161] is a system based on Karaoke that provides more flexibility to users, allows them to use multiple devices and tolerates users going offline to some extent. The tradeoff for this is a higher latency at 32 seconds. The Yodel [162] system achieves voice communications in a mix type system with approximately 1 second of added delay. It, however, achieves this only by working with clients who will be connected to the network a high percentage of the time (phone devices typically remain constantly connected to their network) and by providing weak anonymity guarantees. Talek [163] is a non-mixnet system with stronger anonymity guarantees but is a highly specialised system suited only for simple messaging. Its PIR-based paradigm is also much more computationally expensive for clients. However, it

achieves the best latency levels of any of these systems at 1.7 seconds. All of these systems' assumptions and anonymity guarantees vary, and none can be used as a specific benchmark for the level of delay needed for robust anonymity. However, these examples illustrate that despite intensive effort, robust anonymity from strong adversaries cannot be achieved with a delay as low as that of mainstream anonymity systems like Tor. Tor provides anonymity for communications with between 200ms and 600ms of latency [164]. As stated, this lower level of latency is only achieved due to Tor's lack of protection against stronger adversaries. It is clear that stronger anonymity guarantees still require higher latency than what is used day to day by Tor users. This is the key motivation of the study, which seeks to test the usability achieved at these higher latency levels.

Piotrowska [165] performed a comprehensive evaluation of the Nym Loopix-derived mixnet design and calculated the necessary delay required for the level of anonymity required. The study, however, only considers entropy as a measure of anonymity, which can be a useful metric but does not fully encapsulate the ability of an adversary to deanonymise users of the mixnet. The study shows that high entropy anonymity can be achieved but varies depending on the number of users of the network. Further work must be done to find delay parameters for mixnets, which result in accurate, quantifiable results for anonymity.

As Nym is currently the most widely deployed mixnet, it is particularly interesting. The low levels of added latency used by Nym likely will not provide sufficient levels of anonymity given the most complex adversary attacks. A new style of more complex deanonymisation attack is slowly developing in the literature. In the future, all mixnet systems will likely need to offer increased added latency to combat these attacks.

3.5.6 Potential Improvements in Deanonymisation Attacks

The latest style of mixnet attack has been slowly developing over the past few years. Several examples in the literature show how the more nuanced characteristics of packet flow can be leveraged to deanonymise users more efficiently. I will briefly

discuss some of these examples as a motivator for using mixnets with higher levels of delay.

A study by Ben Guirat *et al.* [166] recently showed that different latency levels could be selected for different applications running over the same mixnet, and this does not reduce the level of anonymity provided by the system and can potentially even make it more difficult to deanonymise users.

The MiXiM [167] system is a method of simulation mixnet design which attempts to take account of a more complex model of communications flow but still does not allow for a full perspective on a potential adversaries ability to deanonymise.

The MixMatch anonymity attack by Oldenburg *et al.* [168] made a recent attempt to analyse mixnet traffic by analysing the flows and patterns of traffic being sent, and not just considering all packets to be independent as has been done in all previous attacks. This advance is an important step in assessing adversaries' full abilities and thus can guide us in selecting Mixnet parameters, making it more difficult for an adversary to deanonymise users of these systems.

A study by Gaballah *et al.* [169] also attempted to perform deanonymising intersection attacks on sample communications data from social media websites. The increased effectiveness of these attacks confirms experimentally the increased attack effectiveness that can be gained by an adversary when studying the patterns of communication flow rather than independent packets.

The parameters selected for the currently deployed Nym mixnet will need to be revised for more complex attacks that may come in the future. Thus, the delay level must be increased dramatically in the future. This forms part of the motivation for examining users' response to higher delay levels in collaborative settings[2].

[2]Hugenroth *et al.* [170] also showed the high power requirements of low latency mixnets, in particular, Nym. Increasing latency levels would make the networks more usable on mobile devices.

3.6 Design of the Study

I aimed to produce an experience that produces the worst usability experience that a user might commonly take part in. Reading the news on a website, for example, would not be a good candidate, as extra delay would make the user wait longer before the web page fully loads and would not directly disrupt their experience. Collaboration, however, requires complex coordination between parties. A disrupted or delayed flow of information between parties can cause major disruption, particularly if users need to regularly adjust their actions depending on what the other users are doing. For example, users filling out a form collaboratively will endeavour not to complete the same sections twice. If information flow is disrupted, users could accidentally fill out the same section twice or have some other form of conflict.

Online collaboration takes many forms, with a variety of different possible tasks and, in particular, a variety of different levels of synchronicity. Some tasks require infrequent interaction between users. For example, a collaborative to-do list may only add changes in the order of minutes or hours, while a shared word processor document can experience changes as frequently as every second. As this study requires scenarios where users are less tolerant of delay, a scenario where collaborating users enter into conflicts often was sought. To facilitate this, the study was designed with a task that allows the worst-case collaborative conflict. Collaborative document editing in a word processor is a common task featuring very rapid changes; user keystrokes at their quickest can occur many times per second, and the nature of large bodies of text means users can potentially make edits very close together, thereby increasing the likelihood of conflict. The task selected for the study should be based on collaborative document editing but also must emphasise scenarios that have the highest possibility of conflict. For example, a scenario where one user edits a single paragraph and a second user edits a second different paragraph would not be appropriate as there is no likelihood of conflict.

I constructed a web application which presents the study participants with a series of general knowledge questions along with free text answer boxes for each question.

Each answer box is designed to mimic a collaborative environment where the data entered can be collaboratively edited. As the participant attempts to answer the questions, a second simulated user will also answer some of the questions, thereby allowing the user to experience how collaboration feels in the environment.

Additional delay will be added to application data as it "travels" between the participant and the simulated user, thereby simulating the delays caused by using a mixnet system. By adjusting delay parameters, the effects of various delays on the user experience can be studied. The question being asked by this study is, while the editor will reach a consistent shared state between users, will it happen seamlessly and quickly enough to avoid hindering the user? And how does this change when the level of delay is varied?

3.6.1 The Simulated Second User

Given that this task simulates collaboration between two or more users, two or more individuals must be able to work on this set of questions simultaneously. Such an experiment with more than two users at once introduces a significant amount of experimental variance. In this case, each instance of the experiment would depend on the participation of different pairs of individuals and would make it considerably more challenging to make inferences based on the results. Therefore, we opt to create a simulated second user. This simulated user will interact with every study participant, filling out questions in a manner that accurately simulates collaborating with another real individual. This approach allows us to maintain consistency in the experimental conditions. It ensures that each participant experiences the same collaborative environment, mitigating the impact of varying user interactions on the results.

In order to ensure a fair simulation, the simulated user is designed to adjust their typing speed dynamically to match the study participant. This is achieved by maintaining a numeric, typing speed comparison count. This count decreases when the second users types, and increases when the study participant types. A large counter indicates that the participant is typing faster than the user, whereas if the counter

drops to 0, this indicates that the second user is typing faster than the study participant.

Maintaining this count allows the system to adjust the simulated users' typing speed to approximately match the study participant. While the counter remains at 0, the simulated users' typing speed is gradually decreased, conversely, while the counter is greater than 10, the simulated users' typing speed is gradually increased.

This system also ensures that if the participant is not typing at all, the simulated user will also stop typing, and activity can resume once the participant begins typing again.

More specifically, at the beginning of the session, the count starts at 10. Whenever the participant types a letter, the counter is incremented by one. Before the simulated user types each letter, the program checks if the count is above 1. If the counter is greater than one, the program types the letter and, with a probability p, decrements the counter by 1. If the count is not positive the user's typing speed is reduced by a constant factor. The count is then rechecked every 2000ms and if the count still remains at 0, the typing speed is reduced again. Conversely, whenever the participant types, the program checks if the count is above 10 and if so, the user's typing speed is increased by a constant factor. The full simulated user algorithm is shown in pseudo code in Listing 3.1

The simulated user is an approximation of a real user. It cannot precisely model how a real user would act. One limitation of this study is that a real user is not used. However, given that introducing a real user also introduces other biases, this was considered the best option. Future work could create a more accurate simulated user, improving the validity of this study's results.

3.6.2 Collaboration Technology

I employed the Automerge protocol and library to provide the collaboration model between the participant and the simulated user. The participant and the simulated user both use their own Automerge data structure, which models the answers to

```
while(true){
    question = selectRandomQuestion()
    if(question.isBlank()){
        if(tokens>0){
            typeCharacter()
            tokens-=1
            delay = sampleDelay()

            wait(200ms)
        } else{
            decreaseTypeSpeed()
            wait(2s)
        }
    }
}
```

Listing 3.1: Simulated user algorithm

each question they are faced with. After every keystroke, the participant's model is updated, and the model is shared with the simulated user. Similarly, whenever the simulated user makes a keystroke, it is shared with the participant.

3.6.3 Delay Levels

The experiment allows the delay level to be varied over different sessions and thus allows users tolerance for different delays to be measured. The study is a within-subjects study design, which causes the same participant to experience multiple levels of delay across multiple trials of the same study scenario. Each participant will complete five different trials. Each trial consists of 14 questions which are randomly sampled from a bank of questions. Each trial is randomly assigned delay level from a list of options.

Delay is added to the study from a Poisson distribution. As discussed, the Loopix mixnet anonymity system provides anonymity based on a parameter to a Poisson distribution. Messages in the anonymity system are delayed in total by a number of samples from the distribution, and so larger parameters to the distribution result in larger delays but also stronger anonymity guarantees. Messages in the study are designed to be fully analogous to this. A sample from a Poisson distribution delays each message between the real participant and the simulated user, and the parameter to that distribution is what is changed for each trial in the study. After

every keystroke by the participant, a sample is taken from the Poisson distribution, and the sending of the keystroke to the simulated user is delayed by the number of milliseconds sampled from the distribution. In this way, collaborating over this web application is like collaborating using the Loopix system.

I adopt advice from Ignat [126] and avoid including delays below average typing speed given from Dhaka [143]. Delays at that level would not provide any significant insight.

The selected delays for testing are as follows: 1,000ms, 4,000ms, 7,000ms and 10,000ms

Several factors inform this selection. Firstly, the lowest delay is above but close to the latency offered by Tor. Tor provides service with between 200 and 600ms [164] under which most types of web activity are considered usable. It would likely not be interesting to study those same delays, as the results would likely show good usability for participants. An added delay of 1 second is greater than, but still on the order of Tor latency[3]. Secondly, the upper limit on the delays being considered is set by the length of the study task itself. In this case, completing all 14 questions was estimated to take on the order of 60 seconds. Any added delay that was more than a small fraction of this task time would cause interference with the task as a whole. If the level of delay matched the length of the task, then on average, messages to and from the second user would not be transmitted by the end of the task. Even a delay of half the length would likely still cause many messages to be lost. Thus, it would not be an accurate evaluation of the delay's effect in general.

Finally, the choice of delay was informed by a previous study of collaborative text editing. As discussed, Ignat *et al.* [126] conducted a user study with delays of 4, 6, 8, and 10 seconds. From their results, these appeared to be options which showed a significant change in usability and thus would be interesting options for my own

[3]As discussed, the delay added to messages in the study follows a Poisson distribution. The delay figures mentioned are the average result of that Poisson distribution. Latency in Tor does not necessarily follow this same distribution. Thus, latency between Tor and this system is not fully comparable

study.

Some literature differentiates between latency and *jitter*. Latency is the delay between packets arriving, and jitter is the level of change there is from the mean latency. Generally, this study does not consider jitter. Latency will vary according to the Poisson distribution, and jitter may vary as latency increases however this is not measured as there is no way for it to be adjusted while sampling from the Poisson distribution.

3.6.4 Out of Order Messages

As a consequence of Poisson sampled delays being introduced, subsequent edits to the text fields may be delayed by different amounts, and thus, their arrival may be out of order. This behaviour is consistent with the Loopix anonymity system, where messages can be reordered in transit, and the application level must tolerate this. In this study, the CRDT data structure provided by Automerge tolerates this without issue. In some circumstances, participants could notice characters appearing in different orders than they were typed by the second user, but the end result will be as intended.

3.6.5 The Questions

The questions being answered by participants are designed to be simple and with relatively obvious answers to most of the population.

A full list of questions is included in the appendix A; however some question-answer samples are included as follows:

- Which month comes after July?, August
- What colour is a banana?, Yellow
- What colour is the sky?, Blue
- What is the name of the planet we live on?, Earth
- What shape has four sides?, Square

- What colour is snow?, White

The intention behind these questions is to provide a task for participants that is easy but requires some cognitive engagement. Some questions deliberately include the answer within the question, for example: "What continent is South Africa in?" other questions include prompts at the end to help participants remember the answer for example: "Which English speaking country is nearby Australia? N_w Z_____" The answer, "New Zealand", should be easily recognisable to most participants, making knowledge of the answer a negligible effect on the results.

Questions are randomised completely, so any more difficult questions have a uniform effect across the trials.

3.6.6 Collecting Results from Participants

One of the key challenges in conducting user evaluations is determining the most effective method to collect data from study participants.

When users are presented with a task, indicators such as the time to complete it or task accuracy can be valuable for assessing their performance. Additionally, specific survey questions can yield insights into users' thoughts and opinions.

For my purposes, all of the testing will be evaluating users' experiences using the custom user study software [171] that was developed. As such, the questionnaires are used to evaluate a user's experience with that software accurately.

I considered literature on designing effective surveys to reduce bias levels and ensure greater quality results. First, I considered the recommendations of Redmiles *et al*. [172], which presents a comprehensive summary of survey design recommendations for security and privacy research. Note that by using the Prolific.com recruitment platform, it is not necessary to ask participants demographic questions directly, as this data has already been collected by Prolific and delivered to us automatically. This avoids biases introduced by these questions. The study also employed pilot or pre-testing to find issues in the survey or sections participants did not understand.

Social desirability bias is also a key consideration for security-related studies where users may report what they believe is the most socially desirable answer and not what they truly would do or believe. This is avoided in the study by using non-self-assessed metrics as the primary result. The time taken to complete the task was used as a metric which cannot be directly biased by participants. Participants are not informed which level of delay is currently being applied, so participants cannot produce a biased result based on how disruptive they think the delay should be.

3.6.6.1 Completion Time

The overall time taken to complete the task is the most crucial metric. Given that the simulated user is working in collaboration with the participant, the task should generally be completed more quickly than if the participant works by themselves. We, therefore, use the time taken as a proxy for "how helpful the simulated user was" and "how well the collaboration went". The quality of collaboration is expected to diminish slowly at first, followed by a sudden and severe drop in user tolerance. If this is the case, it should become apparent in the completion time data. In addition, delay levels which cause the completion time to be significantly lower than the control task (which has no simulated user) are considered to have very poor utility and have been rendered ineffective by the delay level.

Although quantitative measures are the study's main goal, the survey also included some qualitative measurements, which are used to give more context to why the collaboration does or doesn't break down at certain levels. Firstly, the brief free-text question helps provide some context for what went right or wrong. More comprehensively, however, the Automerge library allows us to save the exact text trace for every participant. This means the exact details of the collaboration could be examined in the future, and more nuanced information could be found on the result of the delay.

I did not use the accuracy of the participant's answers as a metric. This would likely not provide any insight as the results will vary depending on the user's knowledge rather than on the success of the collaboration.

3.6.7 Pilot Studies

Several pilot studies were conducted to ensure the study worked as expected on a small number of users. An additional purpose of these pilot studies, however, was to obtain an initial insight into the levels of delay that were best to measure. A priori, it was not clear how quickly the level of user frustration would increase as the delay increased.

3.6.7.1 Learning Effects

Initial studies of 10 participants did not provide results as expected. While studies with a low sample size were not expected to show statistically significant results, these results showed no relation between time taken for the task and delay level. This led me to estimate that the learning effect was having a large impact on the results. Even though the levels of delay were randomised, users were learning how to work with the 2nd user throughout the study, changing the results significantly. To combat this effect and to ensure results were not skewed as the participant was learning how to use the application more efficiently, participants were first introduced to a practice round of the study. The first round the participant takes part in is always a practice round with 5ms of delay added. The round is identical to the others, and the participant is unaware it is a practice round. The results of this round are not included in any of the results calculations.

3.6.7.2 Sequential Questions

After running another pilot study, the results were still not as expected. The next alteration made as a result was a change from randomised question selection by the 2nd user to sequential selection. Initially, I decided that the 2nd user should randomly select questions in the list. This would more realistically model a collaborative situation where users work closely on the same content but not directly editing the same content. This aimed to be analogous to two users editing two sentences at different ends of the same paragraph and not editing the same sentence simultaneously. After seeing the results of the pilot study, I decided to change this so that the user would attempt to answer the next sequential unanswered question

in the list. It was already observed that colleagues who completed the tasks always began at the top and sequentially completed questions instead of in a random fashion. This new sequential approach modelled better a worst-case scenario where conflicts were most likely. The amount of collisions faced is intended to be higher than is seen in most typical applications, but not much higher, approaching a scenario where two users have an unlikely but possible set of collisions. Making this change did appear to produce results that were more expected, although once again, there was insufficient sample size to make any formal statistical determination.

This fix did not, however, provide results that demonstrated the application was working as intended. The comments from participants were again looked at, and it was discovered that participants had developed a strategy for avoiding conflict with the second user. The vast majority of pilot studied participants noticed the pattern of the 2nd user beginning at the top and sequentially completing the questions. As a result, the participants decided to adapt and to begin their question answering from the bottom of the list, this made it so entering into conflict with the 2nd user was very unlikely until the two converged in the middle of the question list. Once again, this was considered a fault in the study as realistic conflict between two users was not being measured. To finally fix this problem it was decided that questions should appear to the user gradually so that conflict would be much more likely between the participant and the 2nd simulated user.

As a final result from the pilot studies, some users reported that the 2nd user was beginning the trial too quickly and that participants had not had a chance to even read the first question before they began. I agreed that having the 2nd user begin immediately when the web page loaded was unrealistic and did not approximate real users. Real users require a few moments to read the first question and think of the answer. To fix this, I added a 3-second delay to the simulated user's initiation. This gives the impression that the 2nd user is reading the question and deciding what to write.

3.6.8 Full Study Description

The study begins with participants reading and completing the information and consent sheet. Participants are briefed on the nature of the study and informed how their information will be used. Participants are also asked for some further demographic information.

Participants are next asked to complete a short understanding check question. It is highlighted to them that they will be completing the survey in collaboration with a simulated user who is not an actual person. It is also again clarified that the purpose of the survey is not to assess users' knowledge of the questions being asked but to see successfully they can collaborate under increased latency. Participants are asked two short questions to ensure that they understand this information. If they do not answer correctly, they are given an additional chance to try again, and if they again do not answer correctly, they are prevented from proceeding and asked to discontinue the survey. This format is required by the Prolific.com platform to fairly exclude participants who have not understood the study sufficiently.

Participants will then move on to the main task. As discussed, this task involves filling out general knowledge questions. While the participant completes the questions provided, a second simulated user begins to enter their responses in some of the answer sections.

Participants experience 6 of these sessions, each with a different random set of questions from the question bank and each with a different random delay from the six options. The first of these six sessions is the practice round. This round appears identical to other rounds to the participant but always occurs with a delay factor of 5ms and always occurs first before any other round. The results of this round are not used in the result calculations. The survey setup is unusual and unlike any task users will likely have to complete daily. As such, they will likely improve at the task very rapidly in their first attempts. The practice round gives users some time to learn the task so that the actual measured tasks 2-6 provide more accurate results. Of the remaining 5 sessions, one is the control session, where a second user does

not assist the participant, and thus, a delay level is not applicable.

3.6.8.1 Question Structure

The 14 questions are not all initially visible to the participant. Initially, only two questions are visible; whenever the answer boxes for all the currently visible questions contain content, another two questions will become visible. This ensures that the participant and second user will be considering two questions at a time and thus increases the probability of collisions. When all 16 questions have appeared, the submit button also becomes available to the participant, and they can submit their answers when they are happy with them. There is no requirement for the answers to be correct before submitting. The order and timing of the answers are randomised.

3.6.8.2 Detailed Simulated User Description

The simulated user maintains a list of the questions that can be asked and the corresponding correct answers. The simulated user, in a sequence, randomly selects questions to fill out and slowly types letters of the correct answer into the corresponding answer box. The sequence is as follows.

1. Randomly select one of the visible questions that has not been answered
2. Type the next letter into the selected box
3. Sample a number from a Poisson Distribution and delay for that amount (mean delay of 700ms)
4. After every letter is typed, check if the current value of the answer box is a substring of the correct answer
5. If not, a conflict has occurred with the participant. Abandon this question and return to 1
6. When the answer is completed, return to 1

The simulated user is designed to ensure a high probability of conflict with the participant, thereby causing the user to experience the full range of inconvenience.

3.6.8.3 Participant Survey

After the participant has completed each question task, they will be debriefed with another survey section, designed to evaluate the user's impressions of the software under the delay experienced. The debriefing questions are shown in Figure 3.1.

This debriefing survey section consists of 2 Likert scale questions asking about the participant's level of frustration and whether the participant felt the second user was allowing them to complete the task faster or slower. There is also a binary question asking if the participants felt they had to adjust their answering strategy to work with the second user effectively. Finally, a free text box is provided where participants can give general comments about their experience with the task.

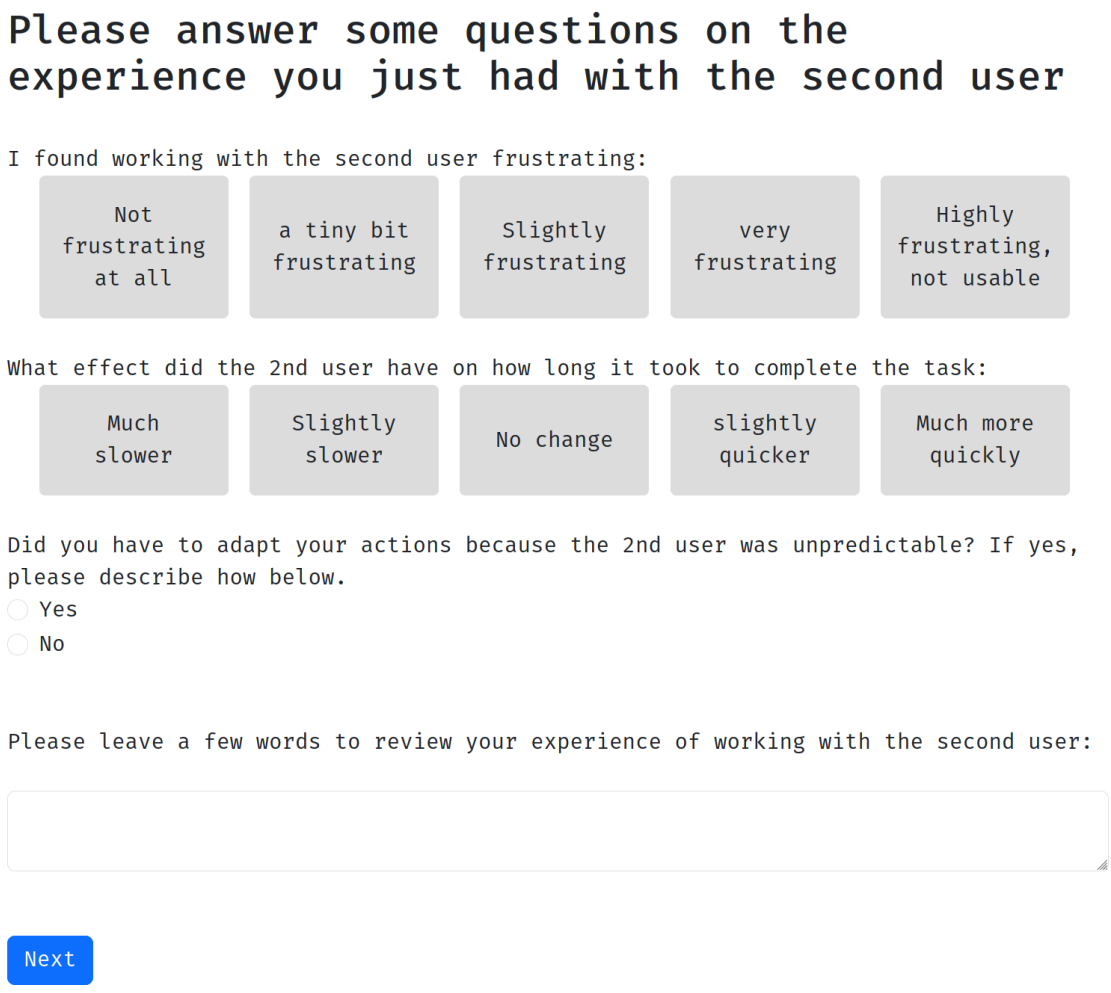


Figure 3.1: The questions given to participants after each task session

These Likert scales were designed based on advice from Redmiles [172]; only five

options were presented to avoid redundant answers, and a neutral term was given, with two positive and two negative terms on either side. Two Likert scales are also presented in reverse order of positivity. The first Likert scale displays its most positive value on the left-hand side, while the second displays its most positive value on the right-hand side.

3.6.9 Recruitment

Recruitment for the study was done through the Prolific.com platform. This allowed us to obtain participants from a wide range of users who signed up for the platform to complete studies and surveys in return for payment. Three hundred participants were recruited to participate in the survey, and the sample was balanced by sex and age, ensuring a somewhat representative sample for the study. Prolific's system provided demographic data for each participant, including age, sex, ethnicity, nationality and employment status. Additionally, participants were asked for their general education level at the start of the study. These demographic data are used primarily to show that the study somewhat represents the UK population. Demographic data can be seen in Table 3.1.

Each participant was paid £1.70 for their participation. The median completion time for participants was 14 minutes and 3 seconds.

Of the 300 recruited participants, 12 responses were invalid due to a software error, leaving 288 participants.

3.7 Results

12 participants were excluded because a technical error invalidated their results.

3.7.1 Completion Times

I initially evaluated the result of changing the delay in the time taken for users to complete the task. I first tested if changing the delay parameter has an overall significant effect on the time taken for tasks. This is done through a Kruskal-Wallis test. I run this test on the time taken for participants under delays of 1000, 4000, 7000 and 10000. The control task (where no assistance was provided) is not included

Table 3.1: Demographics of the study participants

Characteristic	Count	Proportion
Male	139	48.26%
Female	149	51.74%
White	251	87.15%
Asian	13	4.51%
Mixed	5	1.74%
Black	9	3.12%
Other	2	0.69%
Data withheld	8	2.78%
18-29	54	18.75%
30-39	50	17.36%
40-49	45	15.62%
50-64	73	25.35%
65+	66	22.92%

Table 3.2: Education level of the study participants

Education Level	Count	Proportion
Not provided	1	0.35%
None, or incomplete secondary education	2	0.69%
Secondary Education (A levels, GCSE)	115	39.93%
University Degree	110	38.19%
Postgraduate qualification	60	20.83%

in this metric. The resultant test statistic is 60.26 with a p-value of $5E-13$. This extremely low p-value shows there is a statistically significant effect overall on the time it takes participants to complete the question tasks when the delay is altered.

I next attempt to compare the result of each delay with participants having no assistance at all. Firstly, I display the mean completion times of each delay scenario and the no-assistance scenario. These can be seen in Table 3.4. The basic hypothesis that completion time will reduce as delay increases is shown in this table; however, it must also be shown to be statistically significant. This is done using the Games-Howell post-hoc test[4] Each delay parameter was compared directly to the

[4]The Games-Howell test is chosen as the different sets of completion times do not have equal variances. For more information see Games *et al.* [173] or Shingala *et al.* [174].

scenario where the participant had no help from the second user. The results are shown in Table 3.3. It can be see that a participant with a second user interacting with an average delay of 1000ms completes the task significantly quicker than when they have no assistance ($p<0.001$). Again, it is clear in the 4000ms scenario, participants complete the task significantly faster than the control setting. The lower mean difference and the slightly higher p-value of 0.0025 may indicate a lessening of the effect. I did not perform a significance test on the effect between 1000ms and 4000ms [5]. The 7000ms scenario was not significantly different to the no assistance scenario whilst still maintaining a mean difference of 10 115ms in completion time. The failure to find this significant difference shows that it cannot be said that a collaborating user at 7000ms delay cannot necessarily complete their task any quicker than if they were completing the task alone. Finally, the 10 000ms delay scenario was again not significantly different to the no assistance scenario, with an even smaller mean difference of 5549ms and a p-value of 0.9. Overall, it is without doubt that 10 000ms delay results in no discernible difference between completing the task without assistance and completing the task with 7000ms likely results in no discernible difference, or perhaps not one which is large enough to be detected

[5] Performing statistical tests on every possible combination would require high levels of correction to avoid multiple comparisons errors, and doing so would diminish the power of the testing overall

without a larger number of observations.

Table 3.3: Statistical testing of participants completion time versus the level of delay added. Completion times for each delay are tested against the control of the no assistance scenario. The lower delays of 1000ms and 4000ms show significant difference. The higher delays do not show significant difference. The Games-Howell test automatically accounts for multiple testing error and thus no additional correction needs to be done.

Delay (ms)	Mean Difference (ms)	std error	t–value	p–value
1000	−23 612.53	4038.78	4.13	<0.001***
4000	−19 675.11	3787.33	3.67	0.002 49**
7000	−13 729.60	3830.17	2.53	0.085 05
10 000	−5804.74	4762.50	0.86	>0.9

** Significant below 0.01

*** Significant below 0.005

Table 3.4: Mean completion times of the task. A general upward trend can be seen once delay is added, the highest level of delay results in a completion time almost as high as the no assistance scenario.

Delay (ms)	Completion Time (s)
0 (no assist)	101.57
1000	77.96
4000	81.89
7000	87.84
10 000	95.76

To show the effect size that was measured, Cohen's d was calculated for each comparison with the results in Table 3.5. A decreasing effect size is shown for each increase in delay. As expected, the effect size for a delay of 10 000ms is minimal, whilst the smaller delay values result in much larger effect sizes, showing the value of the assistance of the second user only under smaller delay values. I again note, however, that the effects for 7000ms and 10 000ms were not significant, and thus, these effect sizes are merely indicative.

Table 3.5: Size of effect for each delay level when compared with the no assistance scenario.

Delay (ms)	Cohens d
1000	0.34
4000	0.31
7000	0.21
10000	0.07

3.7.2 Likert Scale Evaluation

Our next metric for evaluating users is the responses to the Likert scale questions as asked at the end of every session. At the end of each session, participants were first asked to rate their reaction to the statement: "I found working with the second user frustrating."

The following options were provided.

- Not frustrating at all (1)
- A tiny bit frustrating (2)
- Slightly frustrating (3)
- Very frustrating (4)
- Highly frustrating, not usable (5)

The spread of participant's answers depending on the delay level is illustrated in Figure 3.2. As expected, most participants chose option 1 when they did not receive assistance from a second user, as there was no user to be frustrated with. A general trend can be observed: as the level of delay increases, more participants select higher frustration levels on the questionnaire. In contrast with completion time, which remains high until the delay causes it to plateau, user frustration appears to increase monotonically; thus, there may be no 'sweet spot' for the delay as regards user frustration.

The second question asked of users was, "What effect did the 2nd user have on how

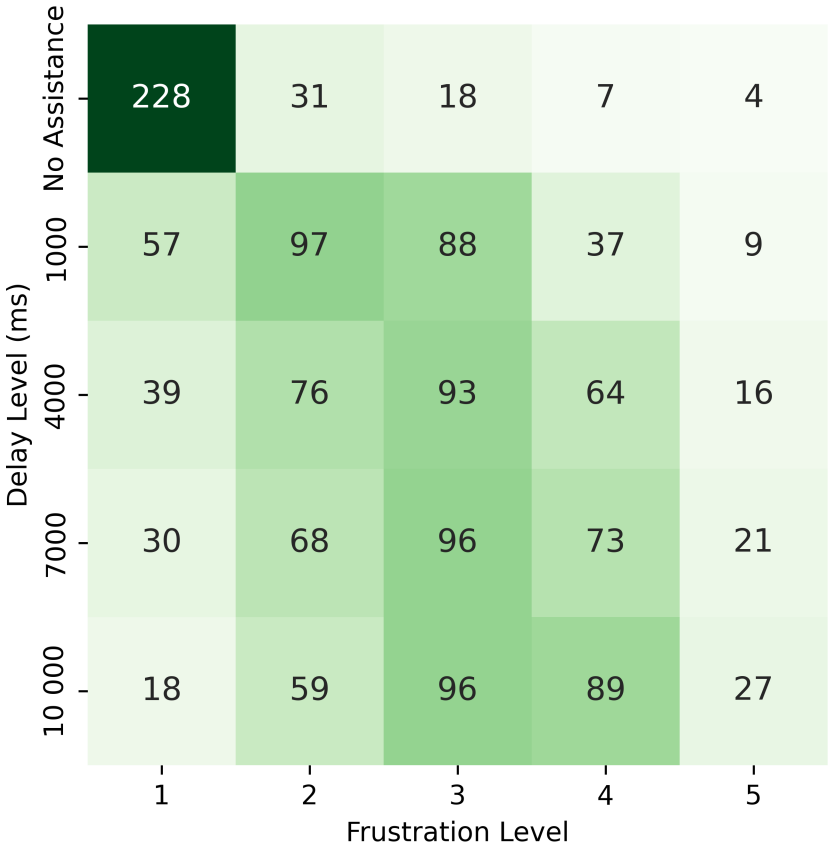


Figure 3.2: Participants reported frustration level depending on the delay. An increasing level of frustration as the delay increases is shown.

long it took to complete the task?" Which aims to assess the speed at which participants thought the second user was helping them to complete the task. Participants could choose a slower option if they perceived the second user slowing them down or a faster option if they felt they were not being hindered and were completing the task quicker with their help. The options are:

- Much slower (-2)
- Slightly slower (-1)
- No change (0)
- slightly quicker (1)

- Much more quickly (2)

The results of this question can be seen in Figure 3.3. A slight trend can be seen here. Participants typically report a more negative effect from the 2nd user as the delay increases. However, the effect is minimal, and the greatest change occurs once the 2nd user is introduced, which is unsurprising. This provides an additional indicator of the previous result, namely that the second user is helpful and increases the participants' productivity until a certain point of delay when the second user is a hindrance rather than a help. This data does not fully comport with the completion time data, however, as some participants believe that the second user is a hindrance even with low delays of 1000ms. This could indicate that although users complete tasks more quickly with assistance, they sometimes perceive it as slower.

Finally, participants were asked: "Did you have to adapt your actions because the 2nd user was unpredictable?" The thought behind this question was that users would have to adopt new strategies to deal with the increased delay, much as was described by Vaghi *et al*. [137]. The results of this question can be seen in Figure 3.4. Overall, the effect is likely minor; there is a large switch from the number of users changing strategy as the 2nd user is added, but the increasing delay from 1000ms to 10 000ms causes a slow increase in the number of users reporting changes of strategy.

3.8 Discussion

The main purpose of this investigation was to provide insight into what kind of average delay could be added to a high-latency anonymity system so that real-time activities could be conducted without totally compromising the utility of the application. From these results, I offer the following basic guidelines for future operations.

3.8.1 Excessive Delay Reduces the Speed of Collaboration

I observed that users' task completion time increases as the average delay increases. More specifically, it was observed that a delay of 10 000ms or more will likely reduce the speed of collaboration such that it is slower than a single user completing the task. The significant fall off of completion time occurred significantly at

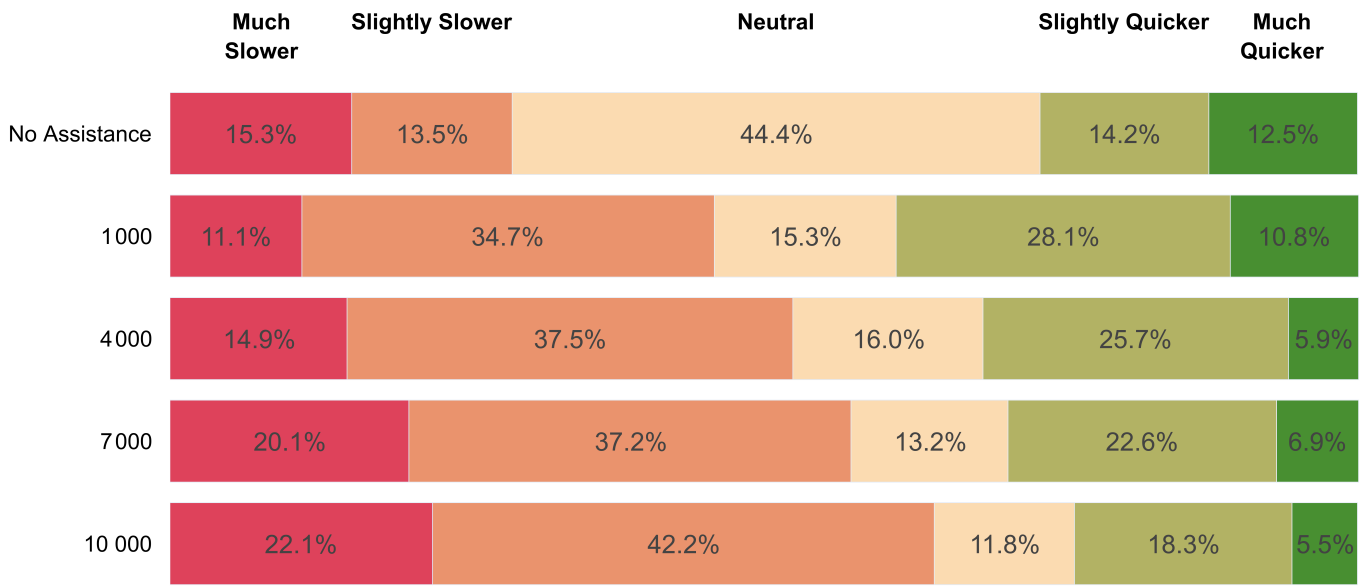


Figure 3.3: Participant reported time effect from simulated user. As the delay increases, more participants believe the second user was slowing down their progress.

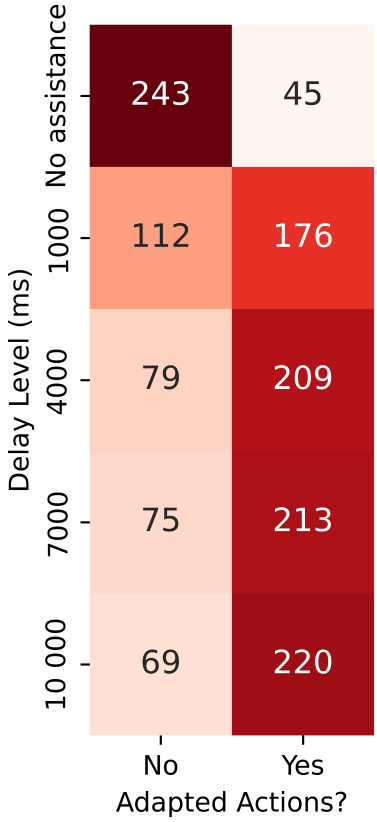


Figure 3.4: Participant reported strategy deviation: Slightly more participants had to change strategy in some way as the delay increased.

10000ms but also seemed to take effect at 7000ms to some degree. This indicates that a mixnet which operates a real-time communication system should avoid these higher delays if completion time is valued by its users. In some scenarios, users working together do so to increase their productivity. Two users may be able to complete the task by themselves but choose to collaborate to speed up the process. In this case, the results show that a delay lower than 7000ms is essential, and even lower delays of perhaps even 1000ms should be considered for maximum efficiency. Other collaborative scenarios, however, are formed of individuals who could not complete the task alone. Where the collaboration is essential to the substance and not the task speed, a higher delay may be permissible; users will be slowed down further, but this can not be seen as inferior to completing the task alone as it would not be possible.

3.8.2 Completion Rate is not Reduced by Increasing Delay

Successful completion of tasks is not reduced by increasing delay, even up to 10000ms. While all of the other recorded variables showed great variation as the average delay varied, the overall completion rate did not change, with only a negligible number of participants abandoning the study. Participants in the study had a small financial incentive of £1.40 to complete the task, which should not have biased them so much as to continue the task in the face of a level of frustration that would cause them to abandon the task in normal circumstances. As a result, one can reasonably estimate that average delays of up to 10000ms will not cause abandonment of tasks and does not cause enough difficulty to make tasks impossible to complete. In situations where strong anonymity is of paramount importance, delay could be increased to 10000ms or potentially beyond and users will likely still be able to complete their task.

3.8.2.1 User Frustration Climbs Steadily with Delay

User frustration climbs steadily as delay increases and may continue to increase beyond 10000ms. Even at lower delays of 4000ms, there was a noted increase in user frustration compared to 1000ms. In contrast to the earlier observations about completion time, even at lower levels of delay, users may become frustrated or have poorer user experiences despite still maintaining good completion times.

3.8.3 Users Often Adopt New Strategies to Assist with Difficult Circumstances

Users often adopt new strategies for dealing with delay, which, depending on the application being used, may vastly improve users' delay tolerance. In the pilot studies discussed earlier, I noted that most participants approached their tasks differently after the practice round. Primarily, this took the form of participants answering questions from the bottom of the list to the top of the list. This strategy dramatically reduced the rate of collisions and conflicts with the second user. It thus negated the reduced completion time for higher delays, as well as the frustration that would normally be associated with it. The study tasks had to be altered to avoid this type

of participant adaptation as it aimed to describe the worst-case scenario for participants' conflicts. It is, therefore, reasonable to expect that depending on the type of application being used, user adaption could dramatically reduce the effects of higher delay. This may occur in unexpected ways, and it is recommended to expect and test for this behaviour of users.

3.8.4 Recommendation to Mixnet Operators

I can, therefore, recommend that when configuring mixnet systems for real-time communications, a good starting point for testing would be between 1000ms and 4000ms. Systems that already use an average delay lower than 1000ms could consider opting for higher delays to bring about higher anonymity guarantees for their users. Depending on the style of communication offered by the system, completion times may not matter as much, and as a result, delays could be pushed much higher than 7000ms. That being said, user satisfaction may diminish dramatically with higher delays. As is clear from the strategy of Tor Browser, user experience is critical for maintaining a large user base of diverse users. If a delay higher than 7000ms is required, it should be rigorously tested for usability. In some applications, the adaptability of users to different scenarios may allow usability to be maintained in the face of higher delay and avoid this 'worst-case scenario' that I have designed.

Given that the most popular current mixnet implementation, Nym, uses much lower levels of added latency, it is hoped that future mixnet systems that aim for more robust levels of anonymity will use the recommendations provided here. As discussed earlier, currently deployed mixnets use an adversarial model that does not take full account of the abilities of the attacker. Recent work has begun to explore these more complex attacks and it is likely that in the future current mixnet configurations will need to be updated to defend against these attacks.

Finally, the work in this chapter can inform systems that do not use deliberately added delay, such as systems that suffer from latency induced by normal network latency. Systems like Tor do not add latency directly, but the latency that results from routing traffic across three different geographically diverse nodes can reach

high levels. This project informs the design of applications in systems like these. In cases where high levels of delay are non-optional, designers of collaborative applications can recommend against using their application with the system. Alternatively, they can add other latency mitigations to their user interface if it is clear that latency will exceed the levels outlined in this work.

3.9 Limitations

3.9.1 Learning Effects

As Vaghi [137] describes, users, when faced with excessive delay in collaborative applications, can develop new strategies to avoid problems. As users complete five separate trials in the study, they may learn similar methods of avoiding conflict. I compensated for this delay by randomising the order that the delays occurred. If users do indeed learn, each delay level will suffer from this bias in equal measure, thereby negating its effect on the results.

3.9.2 Scenario Does not Fully Mimic Reality

The designed scenario of the study attempts to be a good approximation of actual user collaboration, but it is not identical. The task of entering questions as the study is designed does not exactly duplicate any known real-world collaborative application. While I argue that it provides a good approximation of a worst-case collaborative session, there may be details which are left out.

3.9.3 Limited Abilities of Simulated User

The simulated second user that the study participants interact with are also not exact replicas of real users. As Vaghi [137] reminds us, users often develop unique strategies for avoiding conflict in collaborative work. These strategies can involve both users working together to avoid conflict. The simulated user does not allow this and has no way of learning or working together with the participant to avoid conflict. The result from the study shows that a potential maximum tolerable induced delay could indeed be slightly higher due to user strategies such as this.

3.9.4 Limit on Upper Delay

Because of the short nature of the trials used, there is an upper bound on the level of delay that can be measured. In the worst case, as delays reach the length of time it takes for a participant to complete the trial, it will mean that participants complete the trial fully before any interaction is felt from the simulated user. This would not provide any useful data as there will be no interaction between the users. In other cases, even when the delay is not the full length of the trial, the length of time the participant spends not being influenced by interactions with the second user can be significant and is not amortised out properly in a trial with such a short duration.

3.9.5 Does not Account for Larger Groups

Although much collaboration happens in the two-person, one-on-one setting, larger groups are still a prominent form of collaboration. Although I argue that larger groups of over ten people are less common when seeking anonymity, it would still be highly valuable to conduct this study with larger group sizes or between 2 and 5 persons.

3.9.6 Applicable Only to Cases of Reasonably Synchronous Collaboration

I attempted to model a difficult scenario for users of anonymous communications systems. The collaborative setting sought the situation where delays would most affect participants. I also attempted to make participants conflict with the simulated users as much as possible. This is valid when determining a loose lower bound on user experience in that delay, but it does not generalise fully to all scenarios. It is conceivable that users will be vastly more tolerant of delay in other situations like general web browsing. As such, the user study only shows a approximate lower bound on user tolerance.

3.10 Summary

In this chapter, I outlined the problem of delay in the usability of collaborative actions. I described how the study was designed to create a scenario that maximised

the effect of delay on users, which was a collaborative quiz. I discussed how the study was deployed and run. I organised and displayed the study results and made conclusions based on this data. It was shown that delays up to 7000ms could be reasonable for mixnet operations, and results greater than 10 000ms will likely result in collaboration breakdown. Finally, I discussed the implications for mixnet operators based on these results. Overall, this chapter has produced the first modern work showing what kind of delay might be acceptable for users of mixnets for more complex applications and it can potentially feed a discussion for future work bringing mixnets from their previous life as asynchronous communication mechanisms to real-time modern applications.

Chapter 4

HTTPS-Only Modes: what should they aim to do?

> In the year 2023, an Egyptian politician had malware delivered to his phone via MITM when he visited a website that was not using HTTPS.
> This is why we must finish encrypting the goddamn web.
>
> Eva Galperin

4.1 Introduction

HTTPS-Only modes are a web browser mode wherein HTTP sites cannot by default be visited, and users must click through a warning page to visit non-HTTPS sites [175]. This has been an optional mode in the Firefox browser since 2020 [103], as well as in Chrome [96] and Microsoft Edge [105] since 2021.

Over the past number of years, usage of HTTPS on popular sites has become very high. Websites with HTTPS as default rose from 26.9% in 2018 to 84.9% in 2024 [108]. As adoption grows, there has been a clear effort to have more users

use these HTTPS connections and to encourage users not to connect to non-HTTPS sites. We see elements of this beginning, for example, with decisions to block mixed source content in web browsers, for example by Chrome in 2018 [176], and more controversially in 2019 with Google Chrome marking non-HTTPS sites as “insecure" [177]. In the last few years, HTTPS-Only modes have been the latest step by web Browsers to ensure more and more users’ browsing activity is secured by HTTPS.

Measures like these are feasible primarily because of the increased HTTPS adoption on the internet. It is clear that the internet has rapidly been approaching a state of very high HTTPS adoption, although it may still be some time before the long tail of less popular websites adopts HTTPS in larger numbers [122]. This lead me to study the latest evolution of browsers’ anti-HTTP design through HTTPS-Only modes.

The primary function of HTTPS-Only modes is to notify users of potential avoidable man-in-the-middle attacks. HTTPS-Only modes also automatically upgrade connections to HTTPS when available, however, this is a feature that can be implemented without the intrusiveness of a warning page. The primary man-in-the-middle attack that can be conducted is an SSL stripping attack which can surreptitiously remove HTTPS SSL protections. If this attack is performed, a user’s traffic can be subject to eavesdropping or alteration even if their browser normally upgrades all connections to HTTPS. Previously, it would be sufficient to automatically upgrade users’ requests from HTTP to HTTPS. Websites that supported HTTPS would have secure connections to them, and thus the adversary would be defeated in as many cases as possible. With SSL stripping, protections must force connections to be HTTPS when possible, or, if this is not possible, warn users when the site they are visiting does not support HTTPS. This situation is particularly relevant to Tor, where bad exit nodes could intercept traffic. If a bad exit node does attempt an SSL stripping attack, a mechanism like HSTS will cause an unrecoverable error, or, an HTTPS-Only mode warning page can cause a user to rethink their decision to browse to the website.

This thesis contains two projects relating to the design of HTTPS-Only modes in Tor Browser. This first project conducts a qualitative study into HTTPS-Only modes in Tor Browser. There is almost no literature on HTTPS-Only modes, this study serves as one of the first proper investigations into what information should and should not be included on HTTPS-Only mode warning pages. The specific focus of the study was risk in Tor Browser, however, some of the results are also applicable to other web browsers.

The open-ended survey in this project gathered a range of ideas specifically from Tor experts. The survey sought information both about general risks for web browsing without HTTP, as well as risks which are exacerbated when using Tor.

The results of the survey produced a variety of themes which can provide insight into future designs of both Tor browser and other browsers' HTTP-Only Mode warning pages.

4.2 Research Questions

The first stage of the work on HTTPS-Only modes is as mentioned, a survey of the Tor community. The aim of this survey is to gather broad, nonspecific thoughts about HTTPS-Only modes and HTTPS from members of the community. There are three specific goals for this survey; firstly, it is important that future designers are aware of as many possible issues that could come into play for HTTPS-Only modes, be these different risks that users face when they load non-HTTPS websites, different styles of attacks that are possible, or general usability issues that users could face.

The second goal of the survey is to find any details specifically relevant to Tor browser. One particular issue which was raised is the possibility of bad Tor exit nodes. This is a particular worry to users as the potential for bad exit nodes can mean that using non-HTTPS sites can be even more dangerous for Tor users than it might be for users of other browsers, as before, this is another insight that can be incorporated into the design of warning pages.

The final goal of the first survey is to obtain some specific feedback on the currently available HTTPS-Only mode warning pages, showing participants 3 of the currently used warnings has given us some broad insight into how these might be improved. Crucially, this gives us the ability to design some altered version of these pages to use in the next survey. In short, this is the first step in designing a better warning page, which is then evaluated on a broader sample of users to prove its effectiveness.

4.3 Appropriate HTTPS Risk Models

Before surveying members of the community, some conclusions can be drawn about the risk that is involved in visiting non-HTTPS websites.

It is necessarily true that users are more at risk visiting websites which previously implemented HTTPS but no longer do. In this scenario, a web browser user (Tor or otherwise) visits a website which they have visited before. The user remembers that the site supported HTTPS in the past, however, they are faced now with a HTTPS-Only mode warning as the site does not now appear to support HTTPS.

From the study by Mirian [122] it is clear that web administrators rarely disable HTTPS, only 0.05% of the top websites surveyed did not renew their certificates over a 3 month period. Barriers to setting up HTTPS connections were, in the past, often financial. After the advent of Let'sEncrypt, the barriers to setting up HTTPS websites are often technical rather than financial; if a web administrator knows how to setup HTTPS they will do so. For this reason, the rate of HTTPS sites being disabled is low. When a site that previously supported HTTPS now does not report to support HTTPS, the most likely scenario is that an SSL stripping attack has occurred. An attacker is attempting to fool the user into visiting a false or modified website.

4.3.1 Users Should Expect HTTPS to be Available on Popular Websites

My second proposition is that users should have a higher expectation of HTTPS when visiting popular websites. This holds due to the extraordinarily high adoption

rates of HTTPS for the most popular websites. Google currently shows 100 out of 100 of the tops sites being browsed as having HTTPS [115], these top 100 websites account for about 25% of web traffic. Scans from Scott Helme [178] show that approximately 71.6% of the top 1 million websites support HTTPS, and of the top 4000 websites approximately 76% support HTTPS. This means that browsing a popular website and being faced with a HTTPS-Only mode warning has a high chance of being caused by an SSL stripping attack.

These models of users HTTPS expectations will be used in the analysis of the survey results, and also in developing new warning pages in the next thesis chapter. These models will also assist in analysing current HTTPS-Only mode warning pages in this chapter.

4.4 Analysis of Current Warning Pages

I will next continue to an analysis of the currently deployed HTTPS-Only mode warning pages from a number of popular web browsers. Surveying the features of these warning pages provided insight into how improved warning pages could be designed. Overall however, I found flaws in all of these current warning pages and thus they did not directly contribute to the newly developed warnings.

4.4.1 Google Chrome

The HTTPS-Only mode warning page included in Google Chrome as of version 118 is short relative to other browsers' warnings. The warning is displayed in Figure 4.1. It provides two simple statements: that the connection is not secure and that the warning has appeared because the site does not support HTTPS. A link to learn more is also provided[1].

Google Chromes standard insecure loading bar can also be seen in grey text. The grey text contrasts the green text provided for HTTPS websites, and the Red text provided for more severe SSL errors like expired certificates. This standard design

[1] `https://support.google.com/chrome/answer/10468685?visit_id=638329860133962496-1923049630&p=first_mode&rd=1#https-only-mode`

is present even when HTTPS-Only mode is not enabled. It is noteworthy that the more serious 'red' branding is not used, potentially to differentiate a HTTPS-Only warning from a more serious warning which would indicate definite danger.

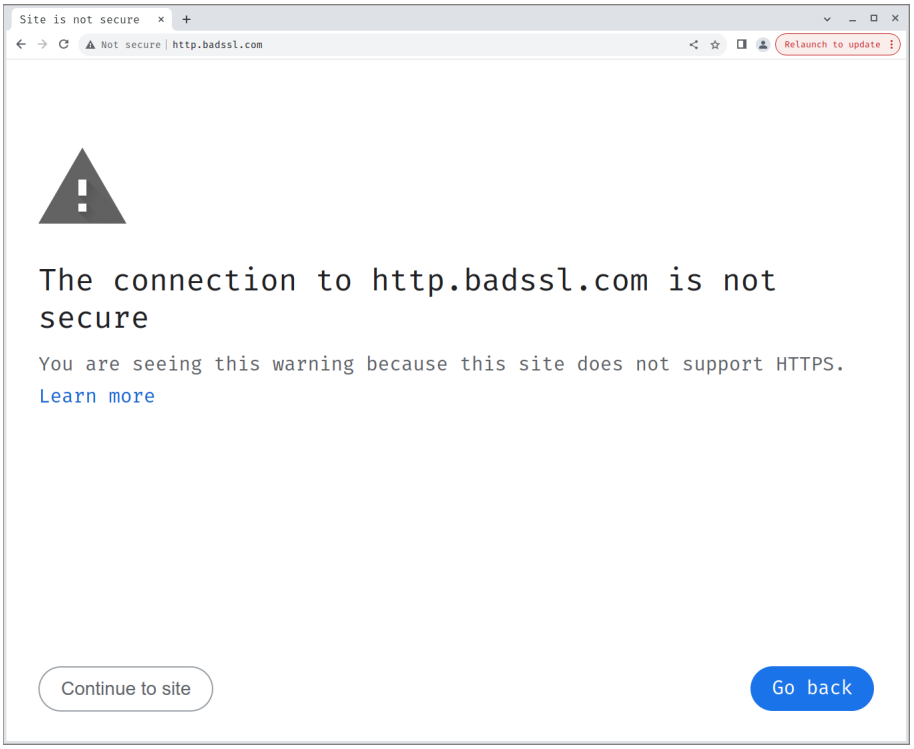


Figure 4.1: Chrome HTTPS-Only mode warning page

4.4.2 Brave

The Brave web browser [179] includes a HTTPS-Only mode [180]. Its behaviour and appearance are identical to that of Google Chrome. Much like Google Chrome, a link is provided for users to learn more[2]

4.4.3 Microsoft Edge

the Microsoft Edge warning contains significantly more information than all other HTTPS-Only mode warning pages as can be seen in Figure 4.2. The information can be divided into three different sections: the initial information block, tips on what to try, and an additional details block, which is hidden until clicked on.

The initial information block declares that the page cannot be reached. The user is

[2] `https://support.brave.com/hc/en-us/articles/15513090104717`

informed that 'Automatic HTTPS' has switched to a HTTPS connection; the user is prompted to try using HTTP instead.

The second block, where tips are given to the user on what to do next recommends 1. Visiting the site with HTTP 2. Checking the network connection, or 3. checking proxy and firewall configuration.

The third section provides the same information as the previous section but with slightly more detail.

Overall, the warning page does not appear to warn the user of any actual danger, and it is likely not clear to users that a lack of HTTPS could be a sign of danger or attack. The first tip to users when seeing this page is to fall back to a HTTP connection, which likely encourages users to do exactly that and not to consider the issue any further.

4.4.4 Safari

Safari does not provide an explicit HTTPS-Only mode; however, it does automatically upgrade all requests to HTTPS.

4.4.5 Firefox

The current Firefox HTTPS-Only mode warning page in Figure 4.3 is also shared by Tor Browser. The warning contains a large header, followed by a statement reminding the user that they have turned on HTTPS-Only mode and that the current website does not support HTTPS. Two scenarios are given, informing the user what might be happening to cause the error; either the website does not support HTTPS, or an attacker could be involved. The user is urged not to enter sensitive information if they continue.

Some changes to the standard text have been made in previous years [181] but no widespread survey or testing has been conducted on these texts.

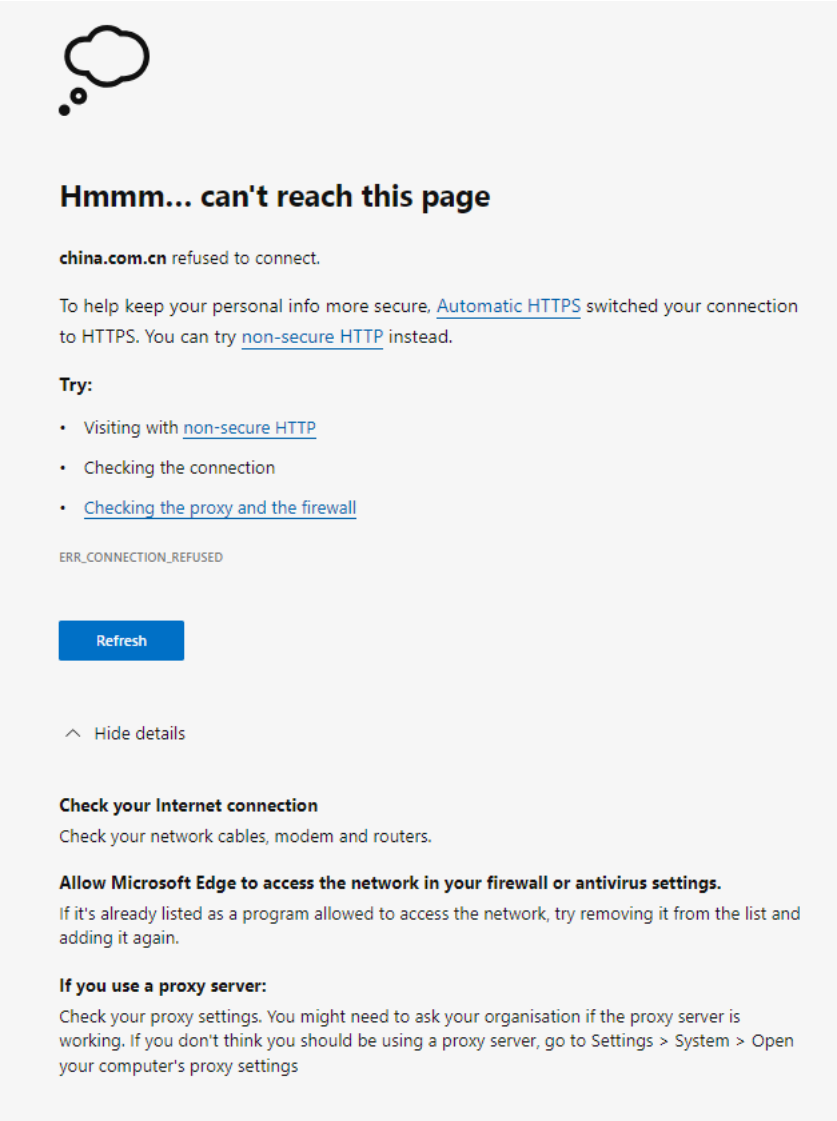


Figure 4.2: Edge HTTPS-Only mode warning page

4.4.6 HTTPS Everywhere (EASE Mode)

The HTTPS Everywhere browser addon has long provided 'EASE Mode' (Encrypt All Sites Eligible) which warns users when connecting to non-HTTPS websites. This warning can be seen in Figure 4.4 This warning page is substantially different to the other warnings discussed. In particular, the information is contained effectively in one large paragraph, in contrast to Firefox and Edge's bullet points or Google Chrome's short length. The warning additionally contains little advice to users, and does not describe any specific warnings or risk scenarios

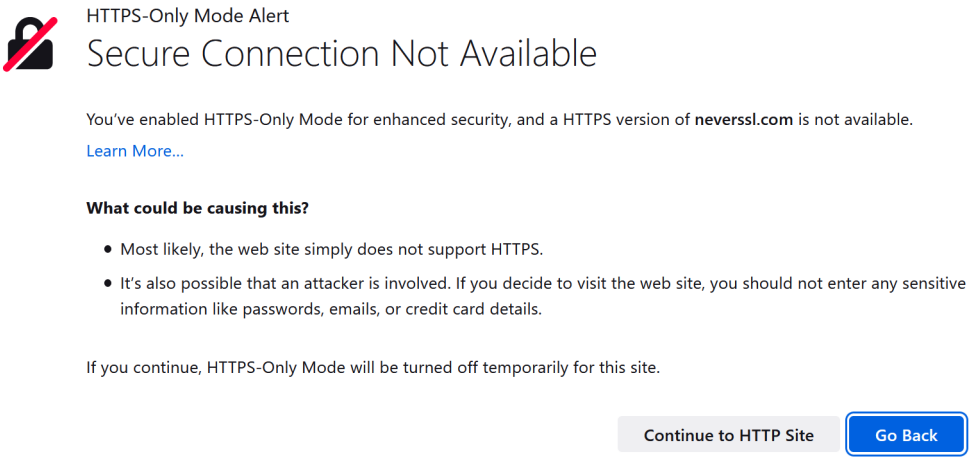


Figure 4.3: Firefox HTTPS-Only mode warning page

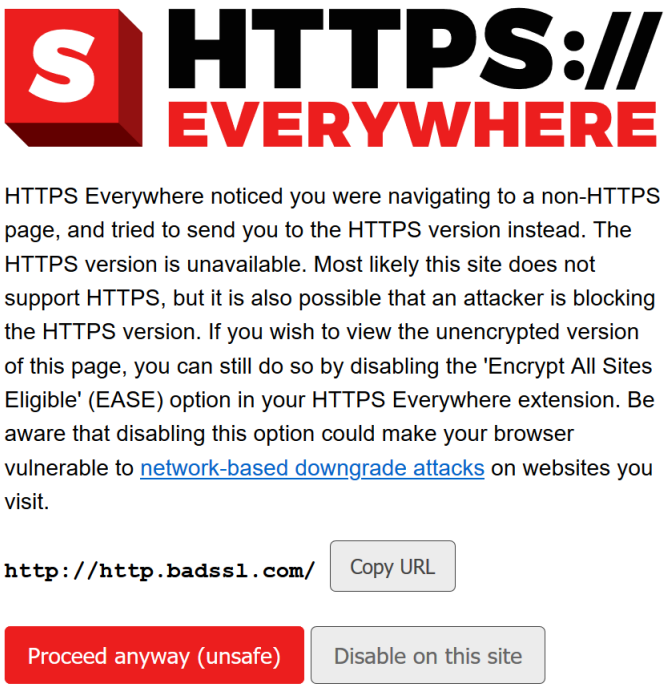


Figure 4.4: HTTPS Everywhere (EASE Mode) warning page

4.5 Survey design

4.5.1 Recruitment

As stated, the goal of this survey is specifically to consult individuals who are already knowledgeable about Tor Browser and are familiar with browser security. To this end, participants were recruited directly from the Tor community. Our three main avenues of recruitment were a recruitment message sent to the official Tor mailing list, a post on the Tor forums, and a tweet sent out by the Tor project Twitter account. Our target number of participants was approximately 30 people, and the final number of responses was 31. Our goal was to have a smaller number of higher quality participant responses. Our survey consisted mainly of open-ended text questions, and the study aimed to encourage participants to be as verbose as they could be and to include any thoughts they had on the subject.

We opted not to make any further reminders or prompts on Twitter or the forums to avoid unethical coercion of participants, as described by Schirmer [182].

Although this was an effective method of finding participants who were exceptionally knowledgeable about Tor and Tor Browser, the participants cannot be guaranteed to be experts. Data from these participants was assessed to ensure that participants with incorrect or incomplete knowledge of Tor were not counted in the data.

4.5.2 Questions

The first section of the survey asks participants a range of broad questions on different aspects of HTTPS, Tor Browser and the different levels of risk involved.

1. While browsing the web, what broad risks do you think occur for a user visiting non-HTTPS websites?

2. Does the level of risk vary depending on the activity being performed? Why do you believe this is the case?

3. Is this level of risk greater for users of Tor Browser, compared to those connecting directly to the site? Or are there new additional risks to visiting non-HTTPS sites when using Tor Browser? What if any, are these risks?

4. Can you identify specific criteria which makes visiting a non-HTTPS website more or less risky? In what scenarios are visiting non-HTTPS sites safe?

5. For you personally, when faced with a non-HTTPS website, what factors affect whether you will continue to the website, or abandon it?

6. Can you comment on what technical factors a user must understand in order to make an informed decision regarding non-HTTPS sites being 'safe' or less risky, and why do you think these are important?

7. TLS provides confidentiality of communications, as well as authenticity and integrity. Do you think these concepts, and how they are provided by TLS is well understood by typical Tor Browser users?

8. Do you believe it is necessary to understand confidentiality, authenticity and integrity of HTTP communications in order to make informed decisions about visiting non-HTTPS websites over Tor Browser? Why do you think this is the case?

The purpose of these initial questions is to gather information about participants' general perceptions of HTTPS. The first question gathers the broad risk factors for browsing non-HTTPS sites. This question is intended to be broad to capture any possible risks or attacks that could occur under any situation when browsing a non-HTTPS site. Identifying these broad risks gives us a starting point to identify what risks HTTPS-Only modes can protect from.

The second question probes users on the varying levels of risk depending on the type of website being visited. It is generally clear that browsing non-HTTPS websites which request a lot of personal information would be riskier than browsing sites where no information is transmitted. It is important however to gather wider views

on this topic and to capture more nuanced views on this concept. If a HTTPS-Only mode warning page is to warn users to take action based on the type of website being visited, it must be established clearly which types of scenarios are less risky.

The survey next asks for more Tor specific answers. The survey sought to contrast the risks users face with the risks that are faced by Tor Browser users. Once again, the most obvious additional risk that Tor browser users face is from bad exit nodes, however a wider range of opinions is needed in order to capture and contextualise as much of this risk as possible.

Question four is a further drilling down into the specific risk factors for using non-HTTPS sites. The intention of the survey is to draw out as many concepts as possible from participants, asking a more specific version of the same question is intended to draw out further ideas which may have surfaced after dealing with the previous questions.

Question 5 attempts to counteract social desirability bias. Participants may overstate risks and dangers in a hypothetical scenario where they identify general risks, however asking about the personal actions taken by the user can achieve more realistic results. For example, a participant may identify many general risks for users, and state that users should never proceed to non-HTTPS sites. This could be considered an overzealous and non practical approach to web browsing and therefore is less useful. Asking about the participants specific actions nullifies this to some extent and forces them to provide more realistic answers based on their own past actions.

4.5.2.1 Surveying Users' Understanding of Technical Factors

We next pursue questions on the technical comprehension of users. HTTPS is a protocol which provides confidentiality, integrity and authenticity of communications. These three features provide all of the benefits of HTTPS and as such a complete understanding of the benefits HTTPS provide requires an understanding of confidentiality, integrity and authenticity.

The question of comprehension is raised specifically by Felt *et al.* [183] where they discuss whether comprehension of SSL warnings can be improved. While the experiment failed at improving users comprehension of warnings, adherence rate still increased. This prompts the question, do users actually need to understand the warning page they are faced with.

Tor Browser often, but not always has users who are more technically experienced. It is possible that Tor Browser's user base is one which will more readily understand the concepts behind HTTPS-Only modes. That being said, as discussed earlier, Tor Browser has a long history of careful and inclusive design in order to make the browser as accessible as possible to users of all backgrounds.

Question 6 inquires about participants perceived view of Tor Browser users' understanding of these confidentiality, integrity and authenticity. Once again this does not serve to be an explicit and conclusive view of Tor Browser users, but assists in drawing out topics and factors which may not have been previously considered.

Finally, question 7 asks participants if users really need to understand these concepts in order to make well informed decisions. This is intended to inform the level of detail that is required on potentially improved HTTPS-Only mode warning pages.

4.5.2.2 Participant's thoughts on current warning pages

The next section of the survey requested thoughts on three current browser warning pages from participants. For each of the three pages, the same three questions were asked.

1. "Do you believe this page accurately informs the user of the risks involved in continuing to a non-HTTPS webpage?"

2. "If not, what issues are left out of this page?"

3. "Do you believe this page either overstates or understates the dangers of non-HTTPS websites?"

Once again, these questions are exploratory in nature. The survey aims to gather as wide a range of thoughts as possible. The three example warning pages serve as a prompt to encourage participants to provide any possible concepts which have been left unexplained by the warning page. We also ask about the intensity of the warning, encouraging participants to either again provide more issues which have been left out of the warning, or, in case the page is over warning, to identify why this might be the case.

In the review of SSL warning literature I found that over-warning was a concern due to the false positives that often occurred in previous years. In modern browsing, false positive SSL warnings are substantially less frequent. In the case of HTTPS-Only mode warnings, false positive warnings are vastly more likely. That is to say that there are many websites which do not implement HTTPS and are legitimate websites, the warning will be displayed, but the reasonable option is for the user to continue to the website anyway. This makes it crucial to craft a warning page which does not over warn, and stop users from proceeding to websites which are in fact benign. Capturing any thoughts on over warning in these current warning pages will provide guidance on how to ensure it does not happen in improved warning pages.

The selected warning pages to display are:

1. The current Firefox HTTPS-Only mode warning page (as of version 96)

2. The HTTPS Everywhere EASE mode warning page as was previously displayed in Tor Browser

3. A previously used Tor Browser HTTPS-Only mode warning page (version 11)

The previously used Tor Browser warning page is in fact an old version of the Firefox warning page. Tor Browser is a variant of Firefox, and due to the nature of the build process which incorporates Firefox's long term support release, it includes an older version of the warning. The current Firefox warning was selected as it was

likely to be included in Tor Browser in the future. Indeed, some months after the completion of the survey, this warning page became the current Tor Browser warning in version 11.5. HTTPS Everywhere was previously included as an addon in every Tor Browser build. Before the advent of Firefox's HTTPS-Only mode, this addon fulfilled the same task when the "EASE Mode" (Encrypt All Sites Eligible) was enabled.

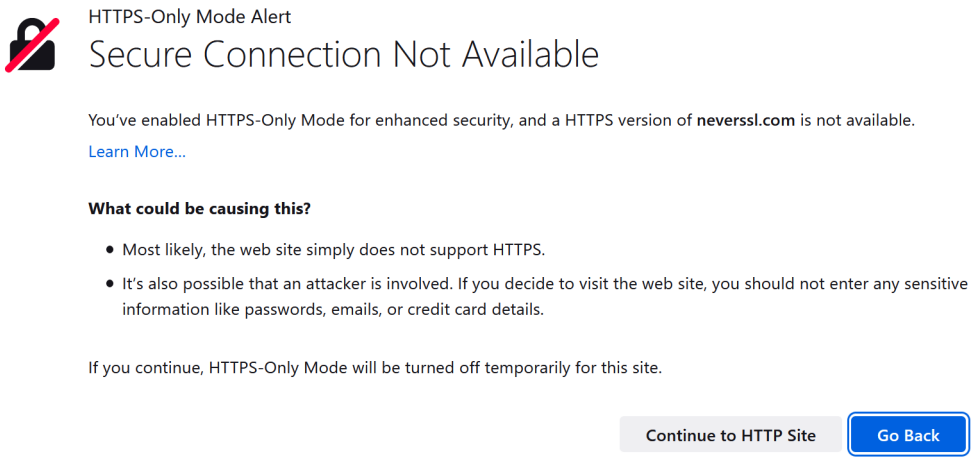


Figure 4.5: Firefox HTTPS-Only mode warning page (version 96)

4.6 Results

Over 4 weeks from 18th January 2022, I received 31 responses to my survey, of which 28 were usable. The three discarded responses were blank, or had very minimal text which was not considered useful to analyse. The mean word count from the used responses was 330. The shortest was 67 words and the longest was 646. This demonstrates the amount of data provided by participants which provided a lengthy insight into their views on HTTPS-Only modes. The data was then organised and coded, providing some quantitative insight into participants' views on the provided HTTPS-Only mode warning pages.

The data was analysed using a Grounded Theory approach. Codes and themes were developed during the analysis process, and preconceptions of themes were de-emphasised.

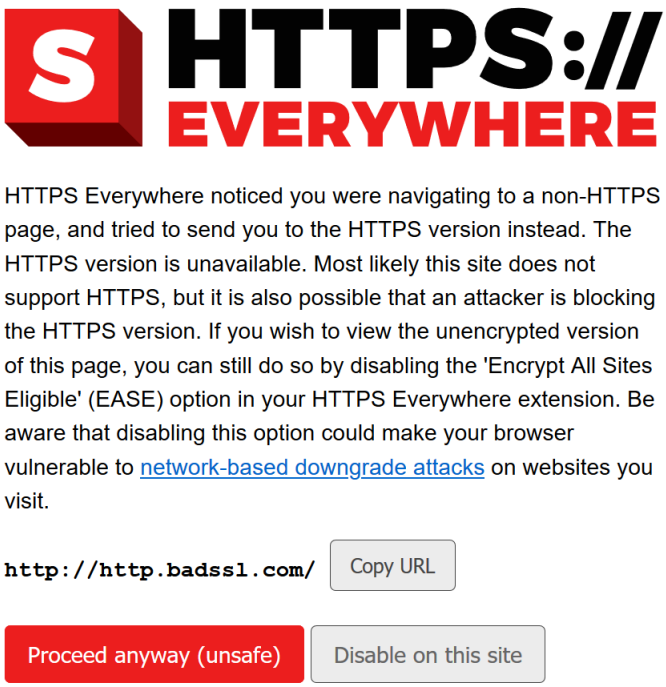


Figure 4.6: HTTPS Everywhere EASE mode warning page as shown in Tor Browse

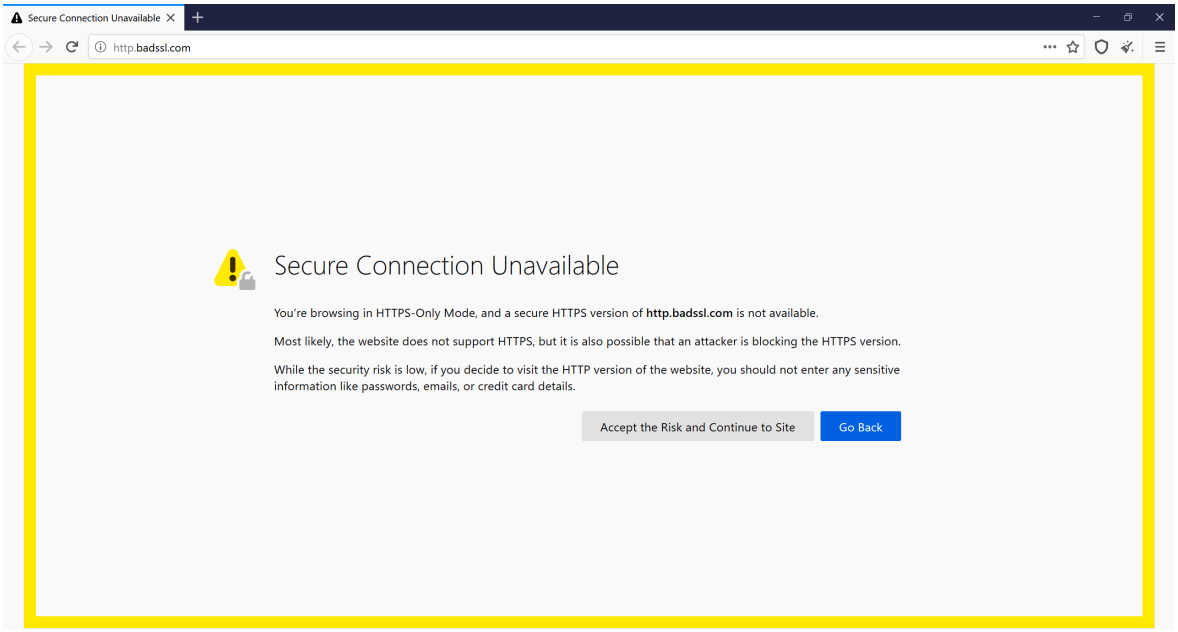


Figure 4.7: Older Tor HTTPS-Only mode warning page (version 11)

4.6.1 General HTTP Risks

4.6.1.1 Logging in Presents Risk

It was broadly agreed by participants that the level of risk varies depending on the activity. One of the most common reasons highlighted was the use of login credentials.

All participants except 5, 9, 13, and 15 agreed that the level of risk varies depending on the activity being performed. This reinforces the view that HTTPS-Only mode warning pages should not by designed like other SSL warnings and should encourage dismissal in some circumstances. Some participants even concluded that the current warnings were too severe and overstated the level of risk to users.

Most participants also agreed that the level of risk is lower for websites which do not require personal information like passwords or payment information P11 stated this succinctly: *"If an important website is HTTP while also holding people's accounts, say a social media site, then data can be easily taken off of the data packets by someone with ill intentions. This will also pose a huge risk of attack as many people use the same few passwords repetitively at many other websites."* This sentiment is also shared by 7 other participants who all noted account login details as a specific risk, which can make non-HTTPS websites more dangerous.

4.6.1.2 Risk for Personal Information

14 participants specifically stated that non-HTTPS websites are less risky if no personal information or account information is being entered. All either stated that there was less risk or no risk for users who did not log in or provide personal information.

P1 goes rather further than this and says that it is only safe if using a public shared computer along with not entering any personal data. This indeed would massively reduce risk to security and privacy, although it is likely not effective security advice as most users will not be using a public computer.

The prevailing opinion is clearly a pragmatic view that notes there is less risk when not entering data, but there is always some risk. As P11 states: *"Say if I am watching youtube/reading a general Wikipedia article (without an account) and let's say that both of these websites are HTTP only. Then there isn't much information revealed about me (other than my IP address and what I am doing there of course). Whereas, if I had an account logged into but it's password didn't match with any other accounts that I had on the internet, then it also shouldn't pose much risk at that specific moment in time."* and as P21 states: *"Websites that don't require the user to provide identifiable data or login info are probably safer, though they might still not be safe depending on the users' threat model."*

P22 conveyed this using the term *"read-only website"* which may be a useful analogy for users in the future. They also further stressed that this reduced risk is not no risk *"Visiting a non-HTTPS site is never fully safe."*.

4.6.1.3 Risk from Downloading Files

One risk which was only mentioned by three participants is the risk of downloading files. When downloading files from sites which are not protected by HTTPS, we lack the authentication and integrity guarantees which would allow us to trust that file. P3 specifically mentions binary files being riskier for users. A binary executable file from an unauthenticated website could have disastrous consequences for security or privacy. Executable files typically have very broad capabilities and this allows any attack a much broader attack surface than they would have with just a static website under their control. I note that this risk factor is not currently discussed in any of the major HTTPS-Only mode warning pages.

Another aspect of the risk of data alteration is noted by P7 who notes that downloading binaries over HTTP is risky. This same risk applies to any downloadable code or script that is run on the user's computer, e.g. Bash or Python. If a user is downloading executable code over a non-HTTPS connection it can be altered by an attacker and thus, if not reviewed properly this code would have a very wide scope for malicious damage to the computer.

4.6.1.4 Risk is Never Zero

Some participants took the opinion that while the level of risk does vary, it is never zero. For some sites, it would be very unwise to proceed without HTTPS, but even when browsing a website where no personal data is entered, and where no level of trust is assumed for the content, there are still risks. P1 discussed JavaScript injection attacks which could affect a user visiting a website which does not have its identity verified. Although JavaScript is typically quite sandboxed inside of modern web browsers, there are many instances where bugs or exploits have been found that allow an attacker to escape this restriction and perform attacks without restriction from within the browser [184, 185, 186]. P16,17 concurred with this view but did not offer any further elaboration.

4.6.1.5 Integrity of the Website

Some participants discussed the risks of data being altered by a third party. The principle of integrity is of course provided by SSL and thus is something that is not available with HTTP connections. P9 states the risk succinctly *"A malicious party can alter the website content."*. P10 then outlines their criteria for deciding if this is a risk for them *"My criteria are whether the plaintext being seen or modified would harm security, and the security status of the browser or computer. Safe example would include visiting a store's website to check their opening hours, or reading a Wikipedia page; it is unlikely that an attacker modifying this information would cause any harm."*

The relative scarcity of this risk in the data compared to the risk to personal data conveys a sense that it is less important, and could indicate lower risk for Tor Browser users. For the most part, risk to users financial data occurs when entering financial information like bank card details online and thus would not fall under this category of risk.

The risk of cryptocurrency attacks as mentioned by participants however is different and can result in a very high risk for cryptocurrency activities on non-HTTPS connections. Cryptocurrency users will often copy the address of another user they

wish to send funds to from a website, browsing to this website does not always involve entering personal information and thus there is no breach of confidentiality for users. A malicious website however could change the data of the website without the user being aware. Changing the cryptocurrency address that was provided would redirect the users funds from the intended recipient to the attacker.

This form of attack could also occur with some bank transfers, although this form of transaction is potentially less common for Tor users due to lack of anonymity as well as the widespread use of Cryptocurrencies to perform transactions on the Tor network. If financial ends are the main motivation for bad exit nodes in Tor, then this scenario of one of the few identified where attackers can directly gain financially by violating the integrity of data.

Unusually however, P22, goes against this and believes that alteration of data is the greatest risk from non-HTTPS websites. *"The worst case is being misinformed, rather than account compromise or financial fraud."* It is not clear why the participant believes this, perhaps it is in the most theoretical sense, if an attacker can change anything you read online, they can conduct significant damage to your outlook on the world, in comparison, financial damage is a known quantity.

4.6.1.6 Injection of Advertising

Participants 1,12,13, and 24 highlighted the risk of injection of advertising. Most responses discuss this risk in the context of Internet Service Providers rather than Tor exit nodes. Both Tor exit nodes and ISPs have the ability to modify unencrypted traffic to insert advertising. P13 claimed direct experience with this: *"I've experienced, first hand, MITM attacks from US companies. They hijack the connection to insert JavaScript powered advertising"*. No participants specifically mentioned seeing this practice performed by Tor exit nodes. I have not found any other mention of this in the literature. It is however, possible that this may occur in the future. Injecting advertising would provide an income for Tor exit node operators who normally volunteer their machines without any payment. Injecting advertising would further contribute to the perverse incentives outlined below in Section 4.6.2.2.

4.6.1.7 Users Should be Aware that Alteration Can Occur in non-HTTP Websites

Opinions from participants were varied on this topic. No one theme appeared to be dominant in the data. Six participants expressed in some form that users should be aware that alterations could be made to the website in question. Some participants note in a general sense that *"the site might not be the one they are accessing"*(p17) or *"The contents of the page might have been modified from what the site tried to send."*(P21).

Other participants propose more specific language for users. P4 provided a reminder that exit node operators specifically can alter the data in their connection. While P9 warns that *"Website content, especially things like bank account number, contact data should not be trusted."*. It is worth noting that these results show participants belief that users will be able to understand this concept and it is reasonable to extrapolate that they believe it should be explained to users.

4.6.2 Does Tor Browser Come with Greater Risk?

Many users mentioned circumstances where Tor Browser users could experience higher levels of risk than other non-Tor browsers. The vast majority of these situations involved bad Tor exit nodes which could act as adversaries against Tor users. This risk from bad Tor exit nodes was persistently mentioned by participants. 17 of the 28 participants directly referenced this risk.

4.6.2.1 Contrast Between Trust Models of ISPs and Tor Exit Nodes

P1 provided a inconclusive opinion on this, comparing the risk of bad exit nodes to the risk of an ISP interfering in traffic. *"Exit nodes are in position to attack and they are known to do so in the past, but they are monitored for that matter, and you can pick a different node if you see anything unusual. When connecting directly, your ISP is in the same position, and they are also known to do bad things - ads and censorship at least. You can't also change ISP so quickly, but they have reputation and may tend not to want to lose it."* This was not a view that was mentioned by other participants, although it is a technically valid risk.

When someone is not using Tor to connect to the internet, the users' ISP has broadly the same powers as a Tor exit node to intercept, modify or read non encrypted traffic. Five participants did express concern that ISPs could perform some form on traffic sniffing on unencrypted connections. The question is perhaps one of trust. P4 states a contrary view whilst directly addressing this issue. *"I believe the risk is greater for Tor users because exit nodes have an economic incentive to modify the traffic for many different reasons, like profiting by adding affiliate links to sites, stealing my credentials/phishing, etc. I don't believe this is as likely on my home internet connection because I have a greater trust in my ISP or VPN provider of them not doing this."* These two comments from P1 and P4 show the contrast between trust of ISPs and trust of Tor exit nodes.

Although ISPs may be more trustworthy because they have an economic incentive to maintain the trust of their user base it is also typically difficult to change ISP quickly, in contrast, Tor exit nodes have much less incentive to maintain trust, however they can be changed out with extreme ease. P10 concurs with P4, stating that the effort required for a bad actor to set up a bad exit node is dramatically less than the effort required to set up an ISP.

4.6.2.2 Perverse Incentives for Exit Node Operators

Providing a concrete example of the risk users face from bad Tor exit nodes, P3 mentions the SSL stripping attacks which occurred throughout 2020 in the Tor network [33]. These attacks focused mainly on cryptocurrency based websites and were an effort to steal users cryptocurrency. This highlights attackers' economic motivation for attacks, perhaps users are at greater risk when conducting activities where an attacker could gain real financial benefit. P4 also mentions the ability of exit nodes to insert affiliate links to websites in order to gain a financial benefit.

P9 highlights the risk of law enforcement deterring legitimate exit node operators. If legitimate operators have to constantly fear action from law enforcement, they are less likely to continue operating their exit node, thus bad exit nodes, which do not have to worry about this risk are encouraged.

Other participants provided other worries about the legitimacy of exit nodes and their ability to attack users. P23 notes that as exit nodes are run by volunteers, they may have malicious motives for running the node. P10 noted that there is very limited oversight processes for exit node operators. Malicious exit nodes are typically only discovered if they perform active attacks. An exit node that alters website content may be easily discovered, whereas a node which is simply reading traffic cannot be detected.

4.6.2.3 Demographic Risk

Although the risk of bad exit nodes was the primary concern raised by participants, some respondents were concerned of the demographic risk to users. P8,17 both agreed that users of Tor are more at risk specifically because of the type of person that uses Tor. P8 believed that: *"many people use Tor for sensitive research."*, P8 did not elaborate on this but the implication of this would be that they believe that those accessing sensitive information may be more at risk of being monitored or attacked. P17 simply states that a Tor Browser user is more likely to be under surveillance. Despite this being a potential factor in the risk facing Tor users, it is a non specific risk and is not necessarily limited to a particular browsing scenario.

Users can face this risk through any number of already specified attack vectors. Translation of this finding into a design decision could consist of a general increase in the severity of warning pages and not a specific guideline for users. More specific language may also be possible that warns users that if they are using Tor Browser for particularly sensitive reasons, they may wish to exercise further caution and not proceed to non-HTTPS websites. Only one participant, P6 believed that there was no additional risk for users of Tor.

4.6.3 Heuristic Measures of Risk

A number of participants discussed different heuristics they used for deciding whether a website was risky or not. As opposed to clear cut indicators of risk, these heuristics provide a inexact assistance to users.

P18 provides an interesting perspective into when one can expect non-HTTPS. They cite the example of a small organisation who do not have the proper IT resources required to set up a HTTPS enabled website.

Only one participant, P20 raised the issue of home versus public Wi-Fi networks. The participant is likely referring to non-HTTPS connections which are not over the Tor network as using Tor would negate any risk from a local network adversary. It is worth noting however that the risk from a local network adversary in a non Tor connection is similar to the risk from a malicious exit node in a Tor connection. Both can alter unauthenticated data at will and eavesdrop with ease.

Another four participants discussed the reputation of the website being a factor. Three participants expressed that they would be more likely to continue to the website if they held it in high regard, as P18 stated *"If I know who owns the website, the organization behind it, I might proceed."*

This reasoning does not protect against man-in-the-middle attacks from Tor exit nodes, a high reputation website will not be safer than a low reputation website when this warning appears. The opposite may in fact be true, a low trust, not widely known website may be less likely to have had HTTPS enabled in the first place, and thus if this warning is faced, it may not be an indicator of attack, conversely, popular widely known websites are likely to implement HTTPS and thus this warning could be more likely to represent an actual attack by a Tor exit node or otherwise. This result may point to a confusion in HTTPS-Only modes even amongst expert Tor Browser users.

P10 reports *"whether the site was previously available over HTTPS."* as being a factor for them proceeding past the warning or not. This is the only report in the data to mention this factor. This factor however may be one of the strongest indicators of a man-in-the-middle attack being performed.

4.6.4 Other Interventions

4.6.4.1 Users Should not be Faced with this Decision & Potentially Warned Away

Three participants expressed the view that users should not necessarily be faced with decisions of this nature and that the best decision should be made for them. P24 states: *"I don't think non technical users can really understand the risks because the risks are mostly invisible."*. This comports somewhat with previous design choices by Tor Browser which seek to minimize users' cognitive load as outlined in Section 2.6 however it may be the case that all cognitive load cannot be removed from users in this scenario and users must be informed somewhat about the risks so they can make an informed choice on whether to proceed or not.

Similarly, three participants advocated strongly warning users away from proceeding under any circumstances. P15 states *"A user must avoid a non-HTTPS site at all times except when it is really necessary"* and P17 states *"A good rule of thumb is 'don't do it."'*. This way of thinking does reduce risk for users, however, depending on the current rate of HTTPS adoption across the web this may incur a cost to users of being discouraged out of using a certain number of legitimate websites.

4.6.4.2 Blocking of JavaScript as a Defense

As a defence against some of these risks, P3 and P27 propose blocking JavaScript. JavaScript can be a significant attack vector for malicious sites, and Tor Browser does include the NoScript addon for blocking JavaScript [187]. By default it is not enabled, but changing the security level of Tor Browser from *'Standard'* to *'Safer'* causes JavaScript to be blocked on non-HTTPS websites [188]. Elevating this to *'Safest'* causes all JavaScript to be blocked. One possible adjustment to the HTTPS-Only mode warning page would be to remind users what level of security level they have set in Tor Browser and suggest that they would benefit from increased safety when visiting non-HTTPS websites at a high level.

4.6.4.3 No Clear Answer on Whether Users Need to Understand CIA

There was a clear divide between participants on this topic; the majority (16) believed that understanding *cia* was necessary for users, whilst eight participants thought that it was not needed. In arguing for the point, P3 states *"for example, you shouldn't download a file over HTTP without checking its integrity. So a warning explaining which potential risks can happen is important."*

For users to understand that files being downloaded over HTTP are more risky, there must be an inherent understanding of integrity. Users must understand that an attacker could alter the file in any way and it would not be flagged by their system (the exception to this being signed software). P11 believes that in the specific case of Tor it is crucial. *"Yes it is absolutely needed, because most people use tor to just anonymously browse the web and visiting such non-HTTPS websites could remove all the layers over them which made them anonymous in the first place."* This is one of few responses which highlights that loss of anonymity can be a result from non-HTTPS browsing. If an attacker can browse the content of users requests, it is possible and even likely that the data contains information which will easily deanonymise the user thereby defeating the purpose of using Tor Browser in the first place.

P12 on the other hand, believes it is a necessary understanding for internet browsing in general and not specifically just for Tor Browser. P20 believes that users may be more likely to ignore the warnings if they do not understand the types of risk they are undergoing from a non-HTTPS site. P18 adopts the view that while it would be useful for users to understand the concepts, the design of the browser should be opinionated and only allow users to take safe actions.

Overall this may indicate the need to integrate specific discussion of *cia* into updated warning pages.

4.6.5 Other Factors

4.6.5.1 Overwhelming Need to Access the Site

Another factor which was seen from participants is the expression that sometimes the need to access a website is particularly great, and thus users will continue, even if they were not confident in it's safety. Five different participants expressed this view in some form. P2 states: *"How much I need to access it to complete whatever task brought me to that resource."*. P8 states *"How much I care about the content (ex, was the headline really interesting?)"* this shows that the factors which contribute to proceeding past a warning can go beyond just the level of perceived risk, users can tolerate greater risk if the reason is important [189].

4.6.6 Participants Views on Firefox (Current Tor Browser Warning Page)

There was mixed opinions amongst participants on whether Firefox's current warning page accurately portrays the correct level of risk to users. 11 participants stated that the page did provide accurate information, while 7 disagreed with this. Most participants did not elaborate on their opinions.

4.6.6.1 Specific Tor Risks Should be Mentioned

When asked what was left out of the page, 18 participants provided an answer. 8 of these responses specifically mentioned adding mention of the specific risks which are faced by users. Another two participants stated that it should be indicated that this could be an issue with the exit node being used and users should be guided to change to a new Tor circuit. P15 stated that the page could be improved by mentioning specific risks that are faced when using Tor.

4.6.6.2 No Mention of Content Alteration Risk

One participant mentioned that there is no mention of the risk of content being altered *"It leaves out the risk of an attacker tampering with the content. Downloads could be swapped out with malware"*.

4.6.6.3 Understating Risk

When asked whether the warning page was understating or overstating the risk the opinion of participants was much clearer. 10 of the 25 participants answering clearly indicated that they believed the page as understating the risks, while only 2 participants indicated it was overstating the risk. One participant believed that the page was understating the risks due to the lack of information of Tor specific attacks. From this it is extrapolated that the participant is referring to the lack of information available on bad Tor exit nodes in the warning page.

Another participant believed that the page would indicate to users that there is no risk involved if no personal information is being entered. Once again this could be referring to manipulation of content which is not mentioned on any of the warning pages. One participant who believed that page overstated the warning, said so because *"the dangers are highly dependent on the nature of the visited website."*. This participant seems to believe that the warning page does not accurately inform users about how the risk varies depending on the type of site being visited.

4.6.7 HTTPS-Everywhere

4.6.7.1 Users May not Understand the Warning Page

Participants were equally split on whether the HTTPS Everywhere warning page accurately describes the risks of continuing past the warning. Eleven participants claimed it is accurate, 12 that it is inaccurate. P27 believes the page provides accurate information, however they also warn that all users may not understand the information provided, *"yes it informs accurately but that does not mean all users will necessary appreciate all the risks language such as 'downgrade attacks' is probably opaque to a subset of users"*. P20 complained that the text was confusing and suggested that overall it should be simplified and state only basic facts.

4.6.7.2 Risks Have not been Properly Explained

P17 expresses the view that the HTTPS-Everywhere warning page does not properly explain the risks involved: *"no. It is insufficiently verbose in the potential explanations for an https connection being unavailable."*. P4 similarly believes that

the page is inadequate but for the opposite reasons *"I don't believe it does as I feel like it is too technical."*. This once again calls us to understand how technical the users of Tor Browser are, it is apparent that participants opinion on the adequacy of the warning page is based upon how technical they view Tor Browser users to be. This is a criteria which is widely understood and referenced by Bauer [190].

Ten participants wrote that the effectiveness of the warning page could be improve by citing specific risks that users face, instead of giving broad technical description of the attacks that could occur. P3 summed this up as follows: *"'network-based downgrade attacks' is term that users probably wouldn't understand, so changing that to real examples like: an attacker could steal your login credentials, tamper your download...that would be more easy to understand."*. The general opinion of participants is that these very specific dangers should be specifically mentioned by the warning pages to make things as apparent as possible. This once again echoes previous work on SSL warnings which shows that focusing on simpler language is an effective strategy when designing pages that users will read and take heed of.

Even users who are familiar with technical terms like man-in-the-middle attack or SSL downgrade attack will require additional cognitive work to translate this into the specific risks they face for the website in question. Some of the warning text suggested effectively communicates these specific risks: *"card numbers or login details may be visible to attackers."* (P7) and *"an attacker could steal your login credentials, tamper your download"* (P3). P4 also suggest warning of *"password theft, phishing, adding spyware, etc."*

4.6.7.3 Tor Users Could Have More Understanding of Risks

Tor Browser users are being technically minded and thus warning pages with slightly more technical detail could be justified, however this was not a widespread view and was only expressed by a few participants. In contrast, some participants believe the warning was too length and that users would not want to read the whole warning. P(15,19)

4.6.7.4 Proceed Button

P12 noted that the proceed button should not include the word “unsafe” as it is not necessarily true. The purpose of a HTTPS-Only mode warning page is to convey an accurate sense of risk whilst there are many websites which do not support HTTPS yet are legitimate and are often low risk to interact with. The tarnishing of all non-HTTPS site as “unsafe” by the warning page may be counterproductive.

4.6.7.5 Lack of Contextual Discussion

P16 believed that the risk overall was overstated, but that there was no indication that the nature of the webpage or the type of interaction that were expected on that page should change the users risk evaluation. Given that the theme of varying risk levels occurs frequently in this data, it is evident that any warning page which does not attempt to help users to make context based decisions will necessarily either overwarn or underwarn users about these risks. It is clear that P12 believes this page does overwarn, but only because these contextual risks have not been accounted for. P26 also followed this view, stating: *“it should inform the user about continuing with caution not to give away personal info instead of saying altogether you’re vulnerable.”*.

Of the participants who believed the warning page understated the risks, 1 specifically mentioned Tor Browser usage as something which was left out.

4.6.8 Old Tor Browser Warning page

There is no clear consensus whether the Tor Browser warning page accurately informs users of the risks. 12 participants believed it accurately informed users, but 8 believed it was inaccurate. Some other participants were happy with the warning, but were clear it could be improved: *“it accurately informs the risks but it can be made better”*.

One participant praised the conciseness of the warning page which is an important aspect of producing clear warnings for users.

Another participant who believes the page is not accurate states: *“Its making a*

judgement on the risk with context of the site". This echoes comments that were made about the Firefox warning page noting that the warning page does not inform users how the risk varies depending on the type of website being visited. In this case, it is clear the participant believes the warning page is overwarning the user to some extent by not informing the user that depending on the context, there may be no issue in proceeding to the website.

When asked what was left out of the Tor Browser warning page, only 16 participants provided an answer. 6 of these participants once again believed that the warning page should include more specific risks. Some participants identified a number of risks which could have been described by the page. One participant highlighted risks to privacy and also more disastrous JavaScript issues *"An attack can still monitor what you do and run scripts."*. Three participants highlighted the lack of explanation of the risk of content alteration. *"No explanation about the risks of page content being altered (like banking account number, contact details, downloaded app being replaced with a malicious one)."*.

4.6.8.1 Generally Understated Risk

As with the previous two warning pages, many more participants believed the page understated the risk rather than overstated. Of the 23 participants who provided answers, 10 considered the page to be understating the risk and 1 believed it overstated. The other 12 believed the warning was okay, or did not have strong opinions. Participants were terse in their descriptions but most followed the sentiment of one participant saying the page was *"de-emphasizing the possibility that something might actually be going home."*. Another participant specifically discussed the style and design of the page stating *"It could be designed even better to grab the users attention. If I were to see this, I would barely even read this page..."*

4.7 Discussion

4.7.1 Context is a Key Indicator of Risk from non-HTTPS Websites

In earlier sections of the survey, participants discussed the different risk factors for non-HTTPS websites and in particular what factors influence whether or not they will continue to the website. The strongest indicator of risk for participants was the need to enter personal information. This ranges from basic information like names or addresses to banking information or credit card numbers. This risk was discussed by a large number of respondents, both as a general risk and when asked about the factors that participants considered themselves when visiting non-HTTPS websites. Other contextual risks were also raised by participants.

Some participants mentioned cryptocurrencies as a risk factor. Since cryptocurrencies are typically transferred by reading a users address (potentially from a website). Non-HTTPS requests could have altered addresses and could redirect users' payments. This may mean users should not participate in cryptocurrency activities on non-HTTPS websites.

A small number of participants mentioned the size of the website being accessed as an indicator of risk. This is most succinctly put by P18 *"Sometimes I am accessing a small coop or small organisation website ... and they just don't have https enabled because of lack of IT resources"*. Although users are still vulnerable to all of the same attacks when using smaller websites, it may be the case that a lack of HTTPS is less surprising and is not indicative of an attack being performed. On the contrary, a large enterprises website would be fully expected to implement HTTPS and being faced with a warning there would indicate a high likelihood of an attack being performed against the user.

4.7.2 Should Users be Encouraged not to Proceed

Currently, HTTPS-Only modes attempt to offer a reasonable choice to users when faced with websites that do not support HTTPS. In some cases, it may be acceptable

to proceed past the warning to a non-HTTPS website, and in other cases, it may be dangerous to do so. Some participants argued that this distinction should be disregarded, and that in fact, warning pages should completely discourage proceeding to non-HTTPS websites of any kind.

Always blocking non-HTTPS websites, or designing warnings in a way that always discourages users from proceeding would preclude users from accessing legitimate websites that do not provide HTTPS. It is not clear what the level of disruption this would cause currently; approximately 84.9% of websites currently provide HTTPS [108] which potentially could cover the vast majority of websites visited by users however it is not known whether the distribution of websites visited by Tor users aligns with this figure.

Current SSL warnings follow this paradigm of discouraging users from proceeding in any circumstance. The success of SSL warning pages is measured purely as the percentage of users who do not proceed past the warning. Key work on SSL warnings by Felt [191, 183] only uses this metric. The appearance of false positive SSL warnings occurs only if a server administrator makes an error whilst setting up their SSL configuration, whether by not signing the certificate properly, setting an incorrect server time, not renewing the certificate or myriad other errors. A false positive SSL error warning will only occur due to a misconfiguration by a server administrator, this contrasts completely with a benign HTTPS-Only mode error which is caused by not choosing to set up an SSL configuration. This difference could be used to argue that HTTPS-Only modes should not attempt to block access to insecure pages. Since a lack of HTTPS is not an error as such and cannot be thought of as a failure of configuration: it would be improper to block these pages. The current high rate of HTTPS deployment may nullify this argument due to the low rate of false positives overall.

As the percentage of HTTPS enabled websites grows higher and higher, the argument for discouraging proceeding becomes more powerful. Not only will users see warning less often, but it may be the case that fewer users ever encounter the

warning legitimately. As HTTPS adoption becomes very high, the number of cases where a user is warned away from a legitimate website that does not implement HTTPS is low. Still, there is no data available, and no literature studying the incidence of legitimate non-HTTPS websites as general HTTPS adoption becomes higher. It is known that most users mostly visit the most popular websites; there is ample data showing that website visits follow a Zipfian distribution [192]. This powerlaw shows that the most popular websites receive an extremely disproportionate level of global website visits. This is only true as a general rule, and there is no data on how often users encounter non-HTTPS websites. Further study of this, could potentially make the argument for discouraging users from proceeding past warnings in any scenario stronger. Currently, however, the argument remains a niche opinion in the data of this study.

4.7.3 Improved Warning Pages

The purpose of this survey was to scope out potential factors which could help in designing improved warning pages which are specific to Tor Browser. While analysing the responses to the survey, I identified 5 different general themes which were highlighted multiple times by participants and which could form the basis of an improved warning page.

4.7.3.1 Integrity of Content: Downloads

A number of participants mentioned the integrity of downloaded files, (in particular executable files or binary programs) as being an important aspect of HTTPS. Upon examining the current warning pages provided, participants remarked that this was not warned about to any extent.

Google Chrome has recently included a new measure which aims to address this issue in their browser [193]. This new optional feature, much like HTTPS-Only modes will cause a warning to be displayed when a file download is attempted over a non-HTTPS connection. This feature, if adopted in Tor Browser would alleviate the concerns posed by the survey participants by ensuring users are properly warned when downloading files over an insecure connection that could be prone to

tampering. This would make it unnecessary to include this risk in warning page texts.

This feature does not however protect users from copy and pasting code or scripts from insecure websites. Another similar, but less powerful feature was rolled out by default in 2021 [194] which warns users against mixed content downloads of executable files. Only insecure downloads which are initiated from a secure connection are considered by this feature however.

4.7.3.2 Integrity of Content: Untrustworthy Information

Once again, one of the commonly highlighted risks from non-HTTPS website was not highlighted in any of the warning pages studied. Typically, the most common threat discussed in loss of information like passwords or bank details. The confidentiality aspect of HTTPS is more discussed and perhaps better understood by regular users. Integrity is less featured in warning material and this was highlighted by many survey respondents. P8 stated about the Firefox warning page: *"No explanation about the risks of page content being altered (like banking account number, contact details, downloaded app being replaced with a malicious one)."*.

In future warning pages, it may be helpful to include warnings about the integrity of websites. For example, a warning could include *"It's possible the information on this site has been altered by an attacker"* A sentence like this would inform users of integrity risks. Other survey participants did discuss the economic value of attacks and as such alteration of information on websites could potentially be less lucrative to attackers and thus less frequent in practice.

4.7.3.3 Examples of Specific Risks

One of the strongest themes throughout the survey was the request for more specific risks to be described in the warning pages. The main description of risk in the Firefox page for example is "Its also possible that an attacker is involved.", HTTPS Everywhere as well as the old Tor Browser page state "it is also possible that an attacker is blocking the HTTPS version".

The overall main drive of the three warning pages examined seems to be: its possible an attack is taking place, do not enter sensitive data. This appears to be good advice for users and covers some of the issues with using non-HTTPS pages however all three pages avoid mentioning any specific problems or attacks that could occur if the user continues to that website. As P4 suggested of the HTTPS Everywhere page: *"It should communicate the risks of using an HTTP version, like password theft, phishing, adding spyware, etc."*. P24 also mentions *"page modification, credential stealing, malware shimming"*. These examples of specifically identified risks could be integrated into the warning page. A naive example of this could be "If you continue to this website, an attacker could view and passwords you enter. If an attack is taking place, your machine could be vulnerable to malware". These statements convey the risk of real dangers to users as opposed to the previous generalised statements.

A danger with this approach is however over-warning based on the context. The insightful comment from P25 sums up this concern, when asked if the Firefox warning page was overstating the danger, they replied *"Unknown, since the dangers are highly dependent on the nature of the visited website."*. I feel this comment accurately conveys the difficult of describing real risks to users through the warning pages, warning users of malware from a website is likely to cause them to abandon visiting the website, even when the context indicates that there is little risk of an attack occurring.

4.7.3.4 Contextual Risk Depending on Type of Website

Given that many survey participants highlighted context as an important factor in the safety of non-HTTPS connections, it is reasonable that an improved HTTPS-Only mode warning page could attempt to convey this to users. Such a warning could aim for either a specific or a general warning text. Specific examples could include mentioning the dangers of entering login information or personal details which could be stolen by an adversary. More general warnings could advise users that the danger from non-HTTPS websites will vary depending on the website, and

that more critical web browsing should not be done on non-HTTPS websites.

4.7.3.5 Lack of Discussion of Tor Specific Risks

One of the most cited risk factors for browsing non-HTTPS websites using Tor was the potential for bad Tor exit nodes to either passively listen to, or alter traffic passing through them. As mentioned, without HTTPS malicious exit nodes can view in plaintext all of the traffic passing through them. They can also go further and attempt active attacks by altering the content of the website being returned to the Tor user. These risks are very specific to Tor and as many participants mentioned are not featured in any of the sampled HTTPS-Only mode warning pages.

4.7.4 The Educational Utility of Warning Pages

As discussed, all previous literature on browser warning pages focuses on SSL errors. These are cases where the most common desired action will be to abandon the website and not proceed through the warning. HTTPS-Only mode warnings are different in that proceeding past the warning can be a desirable action in many circumstances. Given that the context of the website as well as the type of action taken on that website are major factors in determining the risk level, the warning page by itself cannot inform the user whether they can proceed safely.

If users are to make the same informed and calculated decisions as described by the survey respondents, the warning pages must seek to educate users in these more nuanced points. Users may still wish to continue to non-HTTPS websites, however they can act in a fashion that reduces their risk. All of the warning pages studied already do this to some extent by warning users not to enter personal information. This changes the purpose of the warning page from stopping users from proceeding, to altering the behaviour of users when they proceed in order for risk to be reduced. As discussed above, current warning pages do not warn users of integrity concerns and data alteration risks. An improved warning page should not discourage users from visiting pages completely because of this risk, but perhaps educate users that there is a higher level of risk from actions like downloading files, or that they may wish to be more sceptical of information on this website. This would cause a cumu-

lative positive effect over time, resulting in users taking safer action on non-HTTPS websites.

4.7.5 Providing Agency to Users

With all security warnings, users can either be vigorously warned away from the danger, or more moderately informed and tasked with making a choice for themselves. Often this is due to the prevalence of false positive warnings. Typical SSL warnings (certificate invalid, expired etc) have low false positive rates, and it is usually pragmatic to encourage users to not proceed through warnings.

Influencing users in this way also comes at a disadvantage to websites or entities which happen to encounter false positive warnings. Google Chrome began showing non-HTTPS websites as 'insecure' in 2018 [177], this has a great security advantage in encouraging HTTPS for websites, it does however, disadvantage websites which cannot deploy HTTPS for whatever reason. The same now applies for HTTPS-Only Warning pages. Some survey participants clearly favour the approach of preventing or strongly discouraging users from proceeding to non-HTTPS websites. Much like Google's decision, a design like this would increase security overall, by discouraging unencrypted connections, however, it would once again disadvantage websites which are unable to deploy HTTPS.

This decision may be justified for standard web browsers where the number of non-HTTPS websites being accessed can be surveyed easily. It is more difficult to justify this decision for Tor Browser where the websites being accessed by users can not be surveyed. The unknown mismatch between websites visited on standard web browsers and over Tor Browser means one cannot be sure what level of non-HTTPS websites are being accessed. The use of Tor Browser for specific situations like whistle-blowing, or anti-censorship could potentially drive up the level of non-HTTPS websites being accessed, and the importance that the Tor Project sees in supporting these activities may diminish the desire to make any substantial changes.

4.8 Limitations

4.8.1 Unclear Sample

This survey aimed to target Tor experts. Individuals who would be quite knowledgeable on Tor, Tor Browser and the functioning of HTTPS. The results seem to show that this was achieved this to some extent, but there was now way to obtain assurances that participants had sufficient expertise for the questions. Participants in the survey did report a limited number of incorrect assumptions about Tor and HTTPS which indicates that other errors could have been made by the participants.

Additionally, the Tor experts sampled in this project are not likely to experience Tor and Tor Browser in the same way as the average Tor user. Given that we are attempting to design better warning pages for all Tor Browser users, there may be a mismatch between what is desired by the Tor experts and the general userbase.

The sample is also non-exhaustive, there may be many more interesting themes that could be drawn out on this topic but the limited scope of my sample may not have achieved this.

For all of these reasons, it is crucial that the data and results of this survey are not taken as an absolute guide to the future of HTTPS-Only mode warning pages in Tor, but as a first step in exploring this new topic. Further study will need to be done to validate the claims made by participants, perhaps by including their guidance in the design of future warning pages. Strong generalised conclusions cannot be drawn from the data acquired and it must form part of a quantitative study to prove its worth.

4.8.2 Lack of knowledge of HTTPS-Only modes

HTTPS-Only modes are a relatively new feature to web browsers. In particular, while the mode was enabled by default in Tor Browser, the survey was conducted before this occurred. As a result, participants in this survey may not have been fully familiar with HTTPS-Only mode before taking the survey, and thus, their views on the warning pages may not be well developed. This was mitigated by explaining

the modes to participants during the survey. Future work could be more effective as users have had time to deal with HTTPS-Only mode in Tor Browser. Prospective survey participants will likely have more developed thoughts on HTTPS-Only modes and provide improved survey responses.

4.9 Summary

This chapter described the current state of HTTPS-Only modes in web browsers, in particular the content of their warning pages. From this, a broad scoping survey was drawn up in an attempt to gain insight into the views of Tor experts on the risks from non-HTTPS connections. I outlined the results of qualitative coding that was performed on the data and discussed the themes that emerged from this. The final discussion provided a number of design factors which could be incorporated into new improved HTTPS-Only mode warning pages.

Chapter 5

Creating New Warning Pages for HTTPS-Only Modes in Tor Browser

5.1 Introduction

The previous chapter arrived at a number of recommendations for designing improved HTTPS-Only mode warning pages in Tor Browser. In this chapter, these recommendations are utilised, and I design three new warning pages for Tor Browser. This chapter begins with a discussion of the literature surrounding warning page design. Primarily, this consists of SSL warning pages, as this is the closest analogue to the HTTPS-Only mode warning page. An analysis of the current HTTPS-Only Warning pages offered by major browsers is provided in Chapter 4, Section 4.4. After fully surveying the current landscape of HTTPS-Only mode warnings, I outline my design of three new warning pages. The new warning pages are designed to improve users' reactions to HTTP-Only warning pages. Ordinarily, security warning pages attempt to dissuade users from dangerous activity. HTTPS-Only mode warning pages are different; they attempt to provide users with enough information to make a decision for themselves, as the web browser will not know whether it would be risky to proceed to the website without HTTPS. Given that HTTPS-Only mode warning pages have not been studied in the literature before, I derive guidelines for when it would and would not be considered appropriate to proceed to

non-HTTPS websites. These criteria enable the design of an user study which evaluates the newly designed warning pages with respect to these safety criteria. This study shows that one of the newly designed warning pages produces statistically significant improvements to users behaviour with respect to certain criteria.

Finally, the study collected qualitative information regarding users' reactions to the warning pages, and this data forms the first known set of qualitative data showing how users perceive the relatively new feature of HTTPS-Only modes.

5.2 Background

With the goal of designing new candidate warning pages for HTTPS-Only modes, I considered and reviewed a broad range of literature on browser warning mechanisms as well as general literature considering how users react to security warnings.

One of the most commonly noticed aspects of user warnings is habituation. Over time, users who see a security warning more often usually become 'habituated' to it and effectively notice the warning less and less as it is seen more. This is a critical problem as the warning can often fail to be useful once a user has become habituated. I consider the literature on this phenomenon and what actions have been taken to minimise it.

Researchers have also attempted to gain insight into users' mental models of browser security. A number of studies have been done that attempt to understand what users perceive HTTPS to be and how it is helping them. Understanding what users think HTTPS is will assist greatly in designing new pages which warn users about the lack of HTTPS

Another related factor in user warnings is comprehension. It is important to ensure, to the greatest extent possible, that users can understand the details of what is being communicated in a warning. It has been noticed that users often do not critically engage in warnings and often simply close the web page producing the warning rather than critically evaluating if they are actually at risk. Often, this results in safe

default behaviour; however, in more complex circumstances, users lose agency and utility due to these kneejerk reactions.

I also considered the development history of specific web browser warnings. SSL error warning pages provide the most similar type of warning that can be seen in browsers. A wide range of literature exists on the development of better SSL warnings over the past 10-15 years and I will draw on this significantly in searching for design improvements.

Other studies have been completed in relation to HTTPS in the browser; some studies considered indicator items like the 'lock icon', which indicates to a user that HTTPS is enabled for the connection. Other studies introduce different new indicator icons attempting to make users more aware of the lack of HTTPS.

A systematic review of web browser security indicators by Jelovčan *et al.* [195] found significant problems with how security warnings are designed, and found that users often have difficulty understanding indicators specifically because of how they are designed. Despite this damming view from the literature, this project still attempts to use whatever indications can be found in order to make these new warning pages candidates more effective at communicating an accurate message to users.

5.2.1 User Habituation to Warnings

5.2.1.1 False Positives Drive Habituation

In a seminal work on the subject by Krol *et al.* [98], users' response to security warnings on PDF downloads is evaluated. Users were asked to try out a 'new' software for downloading and summarising PDFs. Whilst using this tool, one of the PDF downloads would trigger a security warning. Participants' response to the warning was measured, and qualitative data on the participants' thoughts on the warning were collected. The authors employed two different warning styles, one short and generic and one more specific and detailed.

No statistically significant difference was found between users' responses to these warnings, perhaps indicating the text of the message was not an important factor

in users' decisions. The authors also gathered demographic information from participants and made a number of other conclusions. Interestingly, participants who had a higher declared level of computer experience were statistically more likely to disobey the warning and continue to download the potentially risky PDF. They also found that female participants were more likely to obey the warnings. Overall, the authors argue that the content or style of the warning itself had almost no impact on users' decisions and that users mostly relied on their own set of heuristics for deciding whether the task was risky or not.

The authors cite previous work detailing how users evaluate situations based on characteristics like 'the look of a website'. These personal heuristics are not always useful and can often be wrong. A user evaluating the safety of a website based on its look would not detect any risk on a phishing website which exactly mirrored the look of a legitimate website. Perhaps most crucially, the authors believe that the lack of effectiveness of these warnings is caused by habituation and excessive false positive warnings. The particular example given is warnings that in the past were shown to users when downloading any file in web browsers. These warnings had extremely high false positive rates as web browsers displayed them for almost any file that was downloaded. Web Browsers had no mechanism to evaluate whether these files were actually dangerous and so every file downloaded produced a warning. Users therefore, would receive warnings for legitimate files regularly which habituated them to these warnings.

5.2.1.2 Warning Text Matters in Absence of Habituation

In contrast to the work by Krol, Akhawe and Felt [196] published a study only one year later that studies users click-through rate on SSL, malware and phishing warnings in Firefox and Google Chrome. The results show that warning pages do have an effect and disprove Krol's assertion that security warnings are largely ineffective. This new study collected on a large scale, users click through rate by using Chrome and Firefox telemetry frameworks and collected 25,405,944 impressions. The study also recorded the approximate time spend on the warning page for each instance.

Overall, the malware click through rates were 7.2% for Firefox and 23.2% for Chrome, Phishing warnings had rates of 9.1% and 18% . These figures are not directly relevant to HTTPS-Only modes but do lend credence to the authors theory that security warnings are actually effective in modern browsers, it also potentially shows that the different warning page styles in Firefox and Chrome contribute to their difference, and as such it is clear that the text of the warning page has an effective on user actions. The click through rate for Firefox's SSL warnings was 33% and 70% for Google Chrome. The relatively low figure for Firefox indicates that security warnings can be successful and the higher figure for Google Chrome could indicate that once again, the content of the warning is important, and the text and design of Google Chromes warning may be less effective than Mozilla Firefox. We can see that this was later corroborated by Felt *et al.* in 2015 [183] where this high level of Google Chrome click through relative to Firefox was partially rectified by improving the warning design.

As regards time spent on warning pages, the authors show that in Google Chrome, users who do not proceed past the warning spend significantly longer on the warning page before proceeding, this could indicate that users who proceed past warnings spend less time reading the warning, or less time considering the best course of action. A final point from the authors notes that users who faced the most common type of SSL warning clicked through it faster and more often than the rarer errors. This corroborates the work of Krol *et al.* in their conclusion that warnings were too frequent to be effective, and increases confidence in the conclusion that SSL warnings have become more effective as they become less commonly used and have less false positives.

5.2.1.3 Reducing Habituation to Warnings

Bravo-Lillo et. al [197] attempted to design security-decision interface components that users would be less likely to ignore. The authors aimed to craft 'attractors' which are particular user interface elements that draw users attention to critical information. Five inhibitive attractors were designed, these are designs which prevent

users from clicking a potentially unsafe option for a number of seconds when faced with a decision. The authors also designed other visual attractors, which highlighted specific portions of warning dialogues for example using highlighted backgrounds or fading in and out text. The authors found that these inhibitive indicators significantly reduced the rate at which participants in their study completed operations which caused warnings to display. It was also shown that while these designs reduced click through for scenarios which were deliberately designed to look 'risky', the rate of click through for benign scenarios was relatively stable as the treatment was applied. This could indicate that users are reading the warning page more intently and making a decision for themselves rather than simply abandoning the tasks when faced with the warning.

5.2.1.4 Habituation is Confirmed Physiologically

Anderson [99] conducted an experiment which showed decreased brain activity when participants were shown the same security warnings multiple times. This study uniquely displays the concept of habituation which plagues warning designers, from a physiological perspective. The main result of this study is to more acutely display that habituation is a real effect which as strong physical evidence behind it and that is should be taken seriously by the security community.

Extending this work, Anderson *et al.* [101] completed another experiment which firstly re-confirms the results of the first study but additionally studies the effect of polymorphic security warnings. Although polymorphic warnings had been shown to reduce users risk taking behaviour in relation to warnings, it had not been comprehensively shown that they directly reduced habituation effects. The authors produce a range of 12 different polymorphic warnings, and conduct a users study with participants that are under MRI observation. While completing a primary task of investigating web browser extensions, the participants are faced with security warnings. From users' responses to the warnings, the study shows that polymorphic warnings reduce the effect of habituation in the brain when compared with the control group of static security warnings. Overall, the study adds additional credence to using

polymorphic warnings to reduce habituation.

5.2.1.5 Changes in Style are Temporary Fixes

Bohme [198] produced a study on user habituation to warning pages, in particular with reference to EULA style agreements. Bohme argues that users are highly habituated to longer textual, EULA style warnings and are likely to click through them without properly reading or understanding their content. The study designers change subtle factors like changing the acceptance text from 'I accept' to 'I take part' to make the page appear less like a EULA. By conducting this study with a large set of participants, the authors find that the more a warning resembles a EULA the more likely a user is to click through it. The authors advocate for more sparing use of warnings in other to prevent habituation. This is as opposed to changing warning design itself to ensure that habituation is less likely. The authors maintain that changing the style of the warning may result in short term improvement to users' comprehension of the warning, but in the long term, users will just become habituated again to the newer style.

5.2.2 Mental Models of HTTPS Connections

Bravo-Lillo *et al.* [199] conducted a broad study of warnings in 2011. The study consisted of showing a large variety of different computer warnings to study participants and soliciting qualitative responses in an attempt to understand how users perceive these warnings and why they take particular actions when faced with them. The results of the study show participants' reactions to SSL in different scenarios. One persistent myth that is identified is that some users think SSL warnings could be ignored on banking websites because banks have good security practices. This contributes to a body of work showing that many users misunderstand the purposes of SSL, what advantages it brings, and how they should react in its absence.

A more modern work on this topic was completed by Reeder *et al.* in 2018 [200]. Motivating this new study on warning pages, Reeder claims that telemetry data from modern web browsers has given us great new insight into *whether* users are adhering to warning pages in the real world. This study is much improved from

previous studies where participants were studied in labs and were likely greatly influence by the social desirability bias. This new telemetry data that motivated the study does not however tell us *why* users are making these decisions. The study by Reeder brings the Experience Sampling Method to this problem by asking users their thoughts on warnings directly after they have interacted with them.

The study was conducted by creation of a browser addon which was installed by study participants. Occasionally, after interacting with a warning, participants would receive a notification from the addon asking them to fill out some questions. This allowed the authors to receive qualitative data on users' thoughts rather than a simple binary data point indicating whether they proceeded or not. Participants were faced with a variety of questions, starting with "Why did you proceed/not proceed to the website?", "Have you visited the website before?", "Do you have an account on the websites?" and "How much do you trust the website?". Recruitment for the study was complex as the actual rate of encountering warnings was very low, a very large sample was thus necessary to acquire enough data. The sample recruited also had a high skew towards younger and more educated people for the study's Firefox sample, and younger and male for the Google Chrome sample.

5.2.2.1 Reputation

The authors discovered that a large proportion of Firefox users that continued through SSL warnings did so because of site reputation. Many participants commented that they had visited the site before and they trusted it, and therefore did not need to heed the warning. A lesser but still significant number of participants noted that they had proceeded due to the necessity of visiting the site, and despite the warning they still had a task they wished to complete. Other participants also noted plans they had to reduce risk like not entering personal information which in many cases would reduce the risk from visiting the website. The authors note that a large proportion of the reasons given by participants related to their own personal knowledge of the situation, or their own assessment, and were not related to the warning itself. This mirrors the opinion of Krol [98] discussed earlier. Many years

later it may still be the case that users often do not properly read warning pages and merely use them as a prompt to make their own assessment of the situation from their prior knowledge.

Although some participants may have made accurate assessments from this prior knowledge, the authors report that many users proceed past warnings to websites because they have a high level of trust in the website. This is very likely a poor security assessment by the users. In HTTPS-Only modes, users should make their own assessments of warning situations, however warning page designers must ensure these are accurate assessments and in this case, trusting a website because it was previously legitimate is not an effective strategy for avoiding risk. An SSL error, or a HTTPS-Only mode error highlights an issue with the *connection* to the website and not necessarily the websites itself. Trust in a website cannot be extended to the connection to that website, unless SSL is used without error. Nevertheless, compliance with SSL warnings has increased dramatically, and this finding does not necessarily mean users are not compliant with the warning.

Among Google Chrome participants, the authors found a large number of users who were faced with the warning because they had mistyped or misspelled the url, and thus were not visiting the website they intended. This benefit offered by the warning page is likely duplicated in current HTTPS-Only mode pages where users could similarly notice the incorrect url before proceeding. Critically, the authors report that 8.3% of Chrome participants, and 2.2% of Firefox users explicitly decided to downgrade to a HTTP connection in order to avoid the SSL warning. Additionally 11.8% of Chrome users and 3.9% of Firefox users reported that they did not proceed past the warning, but still visited the site, not explicitly stating, but likely implying that they also downgraded to a HTTP connection to avoid the warning. This finding has particular relevance for HTTPS-Only mode warnings as this indicates that a significant number of users retained a poor mental model of TLS and did not recognise that attempting to visit the same website via HTTP is very likely to be less secure than continuing to a website with SSL, even when that SSL connection reports er-

rors. Changing this perception amongst users will be essential to ensuring properly used HTTPS-Only modes, and this task could potentially fall to the warning page used. 2-3% of participants who did not proceed past the warning mentioned doing so because the warning was on a trusted site. This reasoning is a more accurate model of risk as a warning on a trusted site would be more likely to indicate an attack occurring, instead of some kind of misconfiguration.

There was a significant correlation between users who proceeded through the warning and those who already had an account on the website as well as those who had visited the site before, or who had had seen this warning on the site before. The first of these is a further indication of users poor models of HTTPS risk. Users who had an account on the website could potentially have this account compromised by any attack, and so it should not be a reason users should trust that website more or proceed past the warning. On the other hand, users who had seen the same warning before on the website were more likely to proceed which could be a reasonable action to take. Visiting websites which previously did not have SSL errors and now do is likely dangerous, however if a website already previously reported errors and continues to do so, there has probably not been a change in circumstances and an attack by an adversary is less likely to be the cause. Conversely, seeing a warning where there was none before may indicate that an attack has begun, website administrators who have correctly configured TLS in the past are less likely to make a further mistake in the future. This does however not consider errors on the users system for example the incorrect system time. Overall Reeder *et al*. [200] conclude that despite some issues, web browser SSL warnings are effective in changing user action, and that “the low-hanging fruit in browser warning design has largely been addressed."

In contrast to much earlier work where browser warnings were decried as completely ineffective, in part due to very high false positive rates the authors argue that the issues left to address are more nuanced: “Our data do not show evidence of such issues remaining; instead, improving adherence rates may require address-

ing numerous smaller, more contextual issues.". These findings bring the issues of general TLS warnings more in line with the issues faced by HTTPS-Only mode warnings, where specific contextual and nuanced criteria need to be used to evaluate risk and where explicit instructions from the warning page will not be effective.

5.2.2.2 What is HTTPS to Users?

Krombholz [109] completed a study on user mental models of HTTPS which gives us great insight into how users view HTTPS and thus how they will react to warnings about it. The study focused both on users of HTTPS and administrators of HTTPS websites, however I will draw mainly from the user results. The study involves qualitative analysis of data from 18 HTTPS end users in an attempt to describe the mental models of users.

The initial stage of the study consisted of a drawing task which asked participants to draw on paper, a summary of HTTPS involving the different actors present. From the data, the authors constructed a model of HTTPS which incorporated most of the common correct ideas that users communicated. The authors also describe an 'anti-model of HTTPS' which incorporates participants incorrect thoughts about HTTPS. Based on the discussion taken place with the participants during this drawing task, the authors identified and explored a number of themes.

The first theme identified was "Mistrust in HTTPS and Browser Security Indicators". Participants had poor understanding of browser security indicators like the "Green lock", 9 users reported a lack of knowledge about this indicator. The authors felt that this and other security indicators were almost completely absent from their data indicating that users rarely consider these. Users also believed wrongly that HTTPS would protect them against phishing. Most crucially, no participants made any mention of authentication as a benefit of HTTPS. 7 participants also expressed distrust in HTTPS, doubting its ability to protect them. Some users did understand that HTTPS aimed to provide encryption such that other entities could not observe their data, however they also often doubted the ability of HTTPS to provide this properly. One indicator of this is that a number of participants stated that 2fa was

required due to shortcoming in the security of HTTPS. This study has several implications for the design on SSL warning pages and in particular HTTPS-Only mode warning pages.

The evidence that users do not have any model of the authentication or integrity provided by HTTPS indicates a key problem in warning page design. If users do not understand the authentication they are guaranteed in their everyday browsing experience, then they will not appreciate the risk involved in losing this protection. As a specific example, a user using online banking may not realise the dangers of connecting to their bank via a non-HTTPS connection if they do not appreciate the value of authentication to begin with. This may indicate a need to alter warning pages to convey this risk explicitly, rather than with assumptions about users mental models. Informing a user that they are not longer connecting to an authenticated site may be ineffective, instead informing them that there is a risk the website is not who they say they are may be more effective. Secondly, and relatedly, it is clear that a significant number of users underestimate the security of HTTPS and therefore may not appreciate the security that is lost when it is not available on a website. It is difficult to image how this could be rectified through warning pages. Some attempts could be made using language that supports the security of HTTPS for example, "When you visit this website, you will not be protected by HTTPS which provides significant security benefits".

A recent study by Herbert *et al*. [201] brought the misconceptions shown in Krombholz's work and checked their prevalence in 12 countries. The HTTP misconceptions surveyed appeared with some variance across the countries, although all 12 countries still displayed a significant presence of the misconceptions. This study adds further validity to Krombholz's original conclusions.

Overall I suspect however that this issue will have to be addressed through wider web browser design, not just warning pages.

5.2.2.3 Malware & Phishing Warnings

Kirlappos *et al.* [189] studied online phishing attempts and conducted a user study on users' mental models of website trustworthyness. Users were monitored while they were attempting a shopping task, and when they were presented with some less trustworthy websites their reactions were monitored. The authors found key insights into users' methods for evaluating trustworthiness and found crucial flaws in their reasoning. Primarily, users looked for elements of the website to confirm the trustworthiness like the quality of the website design, rather than looking for negative elements which should have called into question their trust, for example, the lack of a Green lock in the URL would have indicated danger. In particular, users were less likely to pay attention to browser warnings for phishing sites if they were presented with a good price on their purchase. Although they did not explicitly ignore the most serious warnings, their poor mental models still lead them to situations where there were more at risk than was necessary.

Egelman [202] found that methods to increase user attention to phishing warnings did not actually improve compliance. The authors conducted a user study of 59 participants, and recruited them under the guise of an email web client usability evaluation. Participants were divided into three separate groups and a different treatment was applied to each. Each of the treatments was either a slight change in the instruction text of a phishing warning, or an aesthetic change to the warning. There was no statistically significant change in the rate of adherence to the warning. The theory proposed in this case is that users' did not obey the warnings at an improved rate because they were not informed of the specific risks, and thus had a poor understanding of the threat model. Participants often thought they were being exposed to different risks, and they made rational arguments for not obeying the warnings under that risk model. A warning that more accurately conveyed the specific risk that was involved would cause users to make safer decisions.

Almuhimedi *et al.* [203] conducted a study on the factors that cause users to proceed through Google malware warnings. The study focuses on browsing history and

finds that users are less likely to proceed through a warning for a website they have never been to before and conversely are sometimes more likely to proceed through a malware warning for a website they have visited in the past. The study pursued two avenues: firstly, an analysis of real Google Chrome warning impressions. The result was that users who had the website being visited in their browser history already were twice as likely to proceed through the malware warning.

This result prompted the authors to investigate further how website reputation affected click-through rates. Study participants were part of one of two categories, each exploring how users responded to warnings on websites with different reputations. The scenarios considered both the destination website and the website which referred the user to that link. Three scenarios were considered by the two groups: High reputation referrer to high reputation destination, high reputation referrer to low reputation destination and low reputation referrer to high reputation website. There was a significant difference between the self-reported click-through rate for high-reputation and low-reputation destination websites. No such effect was detected between high- and low-reputation referrers. This may indicate that users are much more concerned about the website they are about to visit rather than the website which referred them there.

From their qualitative analysis, the authors noticed some participants seemed to get confused between Google Chrome's malware warnings and other SSL warnings. The authors note this as a particular worry due to the high rate of SSL false positives at the time, compared with the low false positive rate of Google's malware warnings. Since the paper's release, the landscape has changed considerably and SSL warning false positives are not nearly as common.

5.2.2.4 Users' Technical Understanding of the Internet

Kang *et al.* [204] conducted a qualitative study which sought insight into users' mental models of how the internet works. The study also compared the results of participants with no computing background and those with a specialised background. Participants with a computing background did seem to have more accurate

and in-depth mental models of the internet, although crucially, this did not seem to result in more secure behaviour online.

Some study participants exhibited similar misjudgements of trust displayed in other studies. Some participants (with both technical and non-technical backgrounds) seemed to equate the trust they had in a company's website with the security of the connection they used to connect with them. This once again highlights a key error that users tend to make in relation to HTTPS security and which likely should be addressed in HTTPS-Only mode warning pages in order for users to make secure decisions.

The authors conclude that due to the internet's increasing complexity, action must be taken so that users can be protected from online threats without understanding the technical details of the system.

Friedman [205] also found that some users had misconceptions about what was making them safe online and found some evidence that users based the security of their connection on the reputation of the website.

5.2.3 User Comprehension of Warnings

Users are faced with security warnings pages all the time online. Many academic studies focus on whether users comply with these warnings of not, but much fewer discuss whether users *understand* the content of these warnings. I will now discuss some of these studies which will aid in creating newer more comprehensible warnings in my own work.

Felt *et al.* [183] present a study, which attempts to create SSL warnings that are well understood by users. Previous work focused purely on convincing users to adhere to SSL warnings and not proceed to the website in question. This study attempts to go further by ensuring users understand what is going on and why it might be dangerous for them to proceed. The hope is that users who properly understand why they are being asked not to proceed will be more likely to adhere to the warning and not continue to a potentially dangerous website.

In crafting this new SSL warning page promoting user understanding, the authors draw on literature from the wider design literature [190, 206, 207, 208] which advise on techniques for increasing the comprehension of warnings. It is advised that warning text should be simple, non-technical, brief and specific. The authors also attempted to provide a clear course of action as recommended by the literature. The authors began by removing all technical terms from the main text of the new warning page, terms like "certificate", "security credential" or, "encrypted". Users can still click an additional "advanced" button to see more technical information if they so wish.

The authors also target a low reading level for the language they are using. They employ the SMOG reading level formula, which evaluates how complex language is, largely by evaluating the number of multi-syllable words. The authors next highlight one of the many trade-offs they encounter in designing this warning page: brevity versus comprehensiveness. Concepts in SSL can be quite lengthy to explain properly; as such, the authors erred on the side of brevity, as explaining the threat model of SSL attacks in full would likely require a lot of text. It was thought that larger amounts of text may cause the user to not read any of the text, and it was preferable that users read a short amount of text, rather than no text. Specific risk description is the next design criterion. The authors attempted to include very specific descriptions of what could happen should the user continue to the website. The proposed page states that "Attackers might be trying to steal your information" with examples of passwords or credit card information.

The authors attempt to design against 'over-warning'. Overstating the risk level involved may harm more accurate warnings and overall reduce user safety online. They also employ 'opinionated design' and 'choice-attractiveness'. Opinionated design involves giving users clear instructions for what to do when the warning appears as recommended by Sunshine *et al.* [208], recommending a clear course of action enhances adherence to warnings. Choice attractiveness is used to make the adherence choice more visually appealing than the less preferred choice. As other

Google services use a blue button to indicate the affirmative choice, the authors style their warning page to use a blue button to adhere to the warning.

Two experiments were conducted to assess the comprehension of users when faced with the newly designed warnings. A separate experiment was then run to test users' adherence to the new warnings.

Participants were evaluated based on their understanding of three aspects: threat source, data risk, and false positives

If participants fully understood the warning, they should have been able to demonstrate an understanding that the attack was occurring on the network level(and not from malware, etc). They should understand what data is at risk, e.g. when visiting a bank, their bank details are at risk. Finally, participants were evaluated on whether they understood how the likelihood of a false positive changed based on the type of site being visited. SSL warnings on banks are far more likely to result from an attack occurring than when visiting a minor restaurant's website, for example. The authors use the example of "watching a movie" to convey this concept.

The authors find no significant improvement in their warning for comprehension of false positives or data risk. There is a minor significant improvement in threat source understanding, but the overall ability of participants to choose the correct answer was still about the same as chance and thus is not useful at all.

5.2.4 Warning Text & Style

5.2.4.1 Formality of Language

Stokes *et al.* [209] conducted a recent study on how the level of language formality and professional style affected security notice compliance. This study focused on having participants grade both the formality and their likeliness to comply with security notices on a range of the most popular websites. The results concur with previous work that finds the formality of language does matter and that increased compliance with security notices can be found with this higher standard of language. The level of detail provided and the preciseness of the language also helps

to increase compliance with notices. There may, however, be some overlap as more formal language often tends to be more detailed; nevertheless, increasing formality tends to be a useful guideline for writing new warnings, the exception being when formality causes the length of the warning to be excessive, which can in turn cause users to stop reading the warning and take an uninformed action.

5.2.4.2 Both Text and Appearance Alter Behaviour

In Felt *et al.* [191], various experiments are conducted in an attempt to improve adherence to HTTPS error warnings in Google Chrome. The study is unique in that the authors had the ability to deploy their techniques directly to the Google Chrome browser. The experiments are run on users in the real environment of everyday browsing, and as such, it would be difficult to doubt the correctness of the results. The experiment consisted of 6 different new warning pages and the current control page being presented to users. The authors used several hypotheses to design each of these new warning pages. Many of these hypotheses are designed specifically to answer the question of why Firefox has a much-increased adherence rate compared to Google Chrome, and as such, some of the treatments borrow techniques directly from Firefox.

The authors note that Firefox had a much higher adherence rate to SSL warnings than Google Chrome. The authors produce a direct replica of the Firefox SSL warning page and bring it to Google Chrome to test the adherence rate for this page. The authors add an additional click box, which is necessary to proceed past the warning. It was hoped that this friction would make users more likely to 'give up' on skipping past the warning and increasing the adherence rate. The authors also craft a warning page that is similar to Firefox's but is designed with Google branding in order to foster feelings of legitimacy. Finally, a page is crafted that contains images of watching persons to the page which could make people adhere to the warning more.

As stated, the authors of this study had the opportunity to deploy this study directly to Google Chrome users. Users did not have to agree to take part in the study,

and data was collected under the auspices of Chromes' opt-in metrics collection. This enabled a large number of participants, 130 754 in total. Each participant was randomly assigned to one of the 7 warning pages, and it was recorded whether or not they proceeded through the warning in that instance.

The key finding of the study is that the visual appearance of the warning page causes almost half of the large difference between the adherence rate of SSL warnings in Firefox and Chrome. Adding images of people watching had little to no impact on the adherence rate, and, as such, this tactic can be abandoned for future warning pages. The authors attempt to account for the difference in rate that is not caused by visual appearance. They note in particular that the Firefox warning does not use technical jargon, identifies actions users can take to mitigate risk, and hides technical details from users. These are all considered good design practices that have been identified in other work, and as such, they may account for Firefox's more successful warning page.

5.2.5 Decision Retention

Whenever a user proceeds through a HTTPS-Only mode warning, a temporary exception will be saved by the browser. Both Google Chrome and Firefox only retain the exception for the current session, meaning that when the browser is closed and reopened, the warning will display again for that web page.

The benefits of longer exception retention are clearly outlined in work by Weinberger and Felt [210]. When considering SSL warnings, the study arrives at 1 week as a reasonable retention time for warning exceptions. Retention that is too short risks showing the warning to users too many times and can introduce habituation. Showing the warning too infrequently limits a user's chance to change their mind about the exception, potentially putting them at further risk.

Longer exception retention times for HTTPS-Only mode warning pages could be advisable in standard browsers, however, given that Tor Browser does not save user data between sessions, there is no way to implement a longer retention time for Tor

Browser.

5.2.6 Summary of Conclusions from Background Literature

Although HTTPS adoption has grown massively, there are still many legitimate non-HTTPS websites available, and when HTTPS-Only modes are enabled, users could face false positive warnings for legitimate websites. The goal of these warning pages is to encourage users to make an informed decision based on certain factors. The work discussed in the background section indicates that this is a difficult task and that depending on the frequency of the warning, as well as other factors, users may choose to continue to or abandon the website in circumstances where that would be considered incorrect. This also indicates that as the adoption of HTTPS continues to rise users will become less habituated to warning pages and the warnings may become more effective in discouraging the visitation of risky non-HTTPS websites. This may, however, increase the rate of users abandoning visits to legitimate websites that do not support HTTPS. Krol's [98] main point that false positives are the most important factor in warning habituation is relevant here; however, it is not a point this study can take any action on.

Without taking action on the number of false positives, the other key action described in the literature to avoid habituation is polymorphic warnings.

5.3 Designing New Warning Pages

In the previous chapter's survey, I identified a number of different avenues and themes for improving warning pages. Three of these themes were selected for inclusion in this study, and thus, three different new warning pages were created. Generally, the improved warning pages were created to conform to these specific themes, and did not mix and match improvement ideas. This enabled the individual improvement strategies to be measured and any results thus could be used to make further improvements on the merits of those specific strategies and not their combined effects.

Although not mentioned nearly as much in the data, the issue of website popularity

is a crucial factor when determining website risk. This is empirically shown to be true by surveys of HTTPS adoption[122]. The more popular a website is the more likely it is to support HTTPS. Therefore, although this theme was not prominent in the previous survey data, its high effectiveness as an indicator of non-HTTPS risk was chosen as the third motivator for new warning page design.

Whilst many themes were derived from the data, one point was discussed the most by that surveys participants, and this was the context of the webpage, or, what user is doing on that particular web page. This theme suggests that users should be warned that whether a web site is dangerous to visit without HTTPS depends most often on what the user will do on that website, thus, the users intentions matter much more than the actual website itself.

Another point which featured in the data was the lack of Tor Browser specific warnings. This theme was selected specifically due to participants claims that Tor users could be faced with increased danger. The fact that no specific Tor warning pages are available motivated the use of this theme to design a new warning page.

More specifically, the 'website popularity' design was created with the intention of informing users that more popular commercial websites would be more expected to support HTTPS, and thus, seeing a warning would be more unusual.

Secondly, the context warning was crafted warning users about the kinds of activity they would complete on the website. This design attempts to convey the extra risk for different kinds of activities, in particular any activity that involves entering personal data, for example, shopping online or using banking.

Finally, the Tor specific warning highlighted the extra risk of HTTP when using Tor. In effect, using Tor does not involve extra risk in any specific area; it is a broad additional risk for any activity that might be risky in a non-Tor scenario. When using Tor, a malicious exit node could perform any of the attacks imagined when designing warnings for non-Tor users; they may just be more common, and the users may be more vulnerable. For this reason, I opted for a simple change to the warning

page, noting that the user may be at additional risk than usual.

All three new warnings used the same heading text and image as the current control warning. All three warnings also altered the subheading text "What could be causing this?" to: "What should I do about this?".

5.3.1 Popularity Warning

The popularity warning page was designed with input from my previous survey in Chapter 4, as well as using the literature discussed in the background. It is shown in Figure5.1. This warning retained the last line of the original Firefox warning, which was shortened, as well as the other headline text. Three sentences were added in favour of the control text.

"If this is a well known, popular site, an attacker is likely involved. Do not visit the website."

This sentence gives the user specific criteria when the site may be riskier. This is in contrast to the control warning, where users are told the site could be risky but are not given guidance on how to evaluate this risk. I then invoke the imperative statement, telling users not to proceed if this is the case. This complies with the advice given by Stokes *et al.* [209], noting that users are more likely to follow advice if it is clear and imperative.

"If this is not a popular, professional website, you are less at risk"

This next sentence provides the alternate scenario: if the site is not a popular business, not deploying HTTPS may not be unusual.

"A website you trust is more likely to be risky, for example a bank or financial institution"

Finally, I incorporate additional context for users, attempting to dispel the common misconception that websites which are more trustworthy can have their SSL warnings ignored. This myth identified by Bravo-lillo *et al.* [199] as well as Reeder *et*

al. [200] is essential to dispel for accurate use of these warning pages.

All of these sentences retain a formal and professional tone as is recommended by Roesner [209].

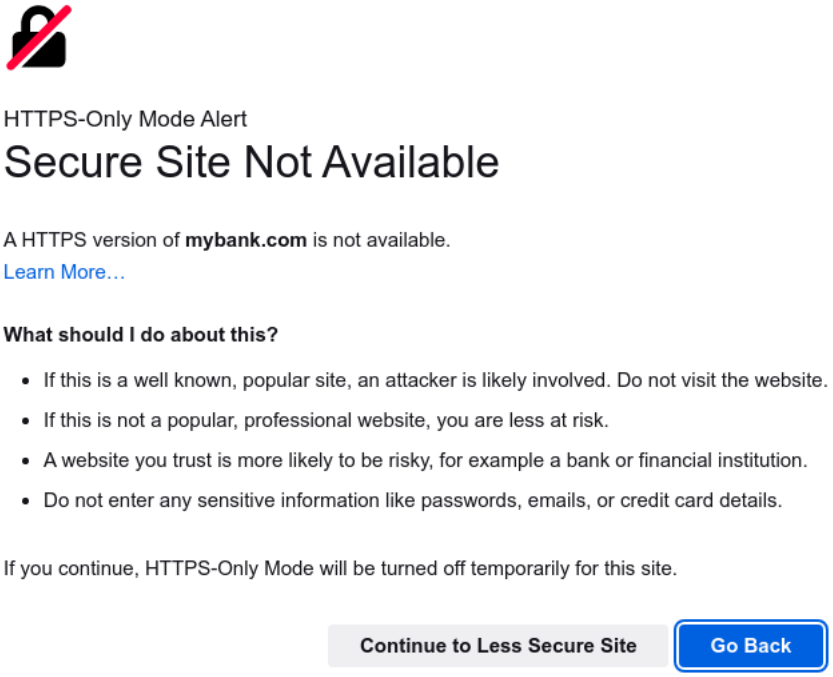


Figure 5.1: Popularity warning

5.3.2 Context Warning

The context warning aims to address more specific concrete risks for users. It is shown in Figure 5.2. My previous survey identified that the biggest concern of all participants was entering private or personal information into non-HTTPS websites. I aimed to address this concern specifically as well as to give users some broader context as to when they would not want to proceed to the website. Although the current Firefox warning page does provide some advice in this category, it is not particularly comprehensive. Participants in the previous chapter's survey consistently highlighted the lack of specific examples of risk, rather than general guidance that 'something might go wrong'. The current Firefox warning states: "you should not

enter any sensitive information like passwords, emails, or credit card details.". My new warning page is an improvement on this and instead states:

"If you visit this site to log in to an account, do shopping, or anything private, you are at risk"

This sentence highlights two of the main risk factors for common web browser usage. Logging into an account is likely one of the most common ways users provide their personal data to websites. Additionally, online shopping will almost always involve entering financial details which could be eavesdropped. I next added the following sentences:

"If you read this site without entering personal details, you are less at risk"

"Any information you enter could be stolen by an attacker"

These sentences once again underscore the risk of entering information that might be stolen, but also that if you are not entering such information there is much less risk.

5.3.3 Tor Warning

It was necessary for the Tor-specific warning to be altered to fit into the context of the survey. The final Tor warning can be seen in Figure 5.3. As I explain in Section 5.5.7, it was not possible to recruit a representative sample of Tor users, and as such, the survey invented a specific scenario which would mimic the dangers faced by Tor. For that reason, this warning page could not specifically mention Tor and instead includes the following phrase:

"If you are using an untrusted network your actions might be monitored"

This sentence attempts to highlight to users the specific danger of using an untrusted connection. Any untrusted network entity could monitor the traffic on this non-HTTPS website. This mirrors the abilities of bad Tor exit nodes to monitor unencrypted traffic.

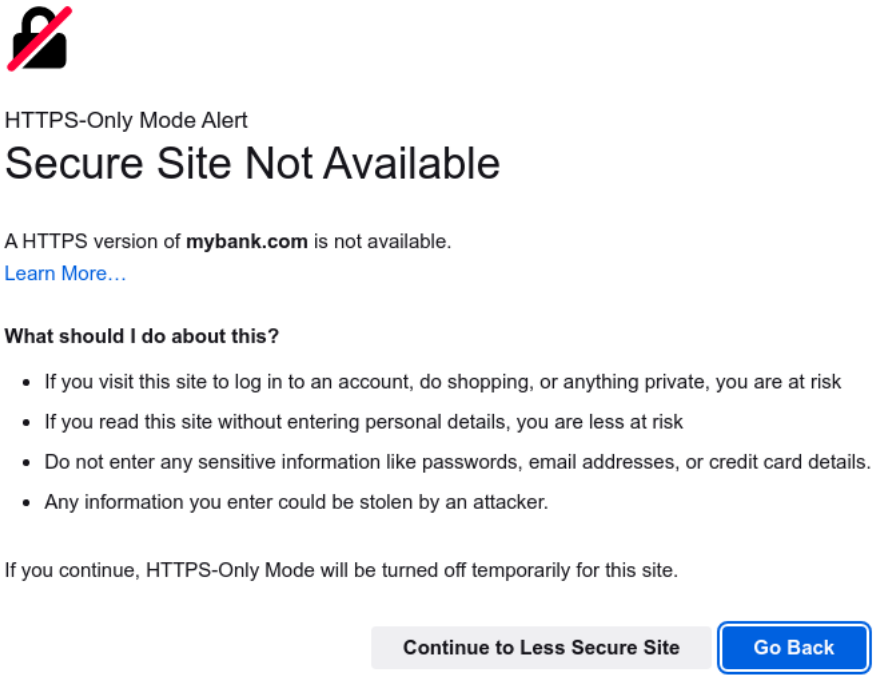


Figure 5.2: Context warning

"If you fully trust you internet connection and any intermediaries, you are less at risk"

This sentence would not be relevant or included in a final version for Tor browser but was included to comport with the situation users were provided with. As users were not told they were supposed to imagine using Tor Browser, it was necessary to include the alternate scenario when the connection was trusted.

5.3.4 Readability

Many factors have been explored here which affect users understanding of, and compliance with, warning pages. One of these factors was the readability of the warning. In Felt's work [183] on improving SSL warnings, the readability of warnings, in particular the 'SMOG' level was evaluated (a higher SMOG level indicates less readability), and changes were implemented that attempted to reduce this measure. Reducing this measure in the new warning pages might be productive, how-

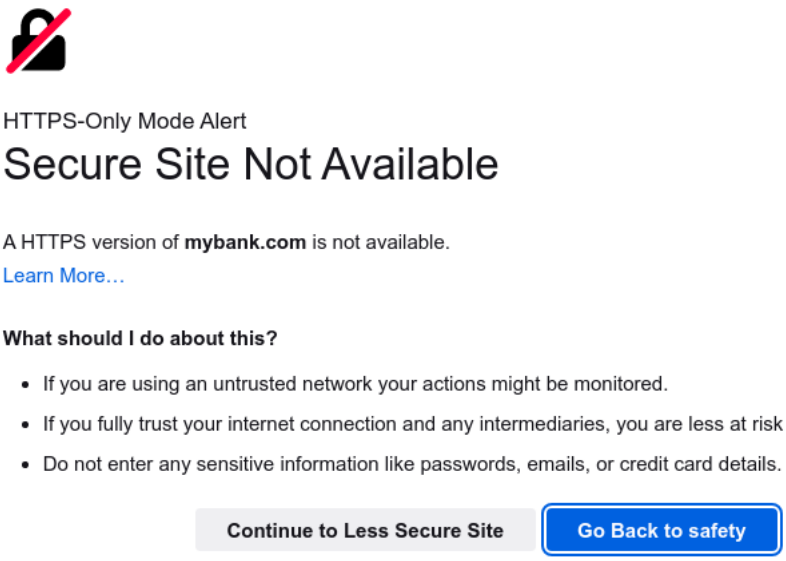


Figure 5.3: Tor warning

ever, making drastic changes to the readability level could overshadow the effect the study is trying to measure. If the new warnings had produced improved responses from users, it would not be clear whether this was caused by improved readability or by the improved warning text. For this reason, the general reading level should ideally remain stable for the new warning pages. The SMOG reading level of each of the new warning pages can be seen in Table 5.1, the new warnings have only slightly increase SMOG scores and are not dramatically different to the control warning in this respect.

Table 5.1: The SMOG reading level of the new warning pages (lower is better)

Warning	SMOG level
Current Firefox	10.13
Popularity	10.69
Context	11.98
Tor-specific	12.16

5.4 Criteria for Proceeding Past the Warning

After crafting these new warning pages, they must next be evaluated to check if the new text produces improved responses from users. To do this, I developed new criteria for what an appropriate response is when faced with an HTTPS-Only mode warning page.

HTTPS-Only mode warning pages are unusual in that there are certain situations when it can be deemed safe, or at least much less risky to proceed past and browse the website without HTTPS. Currently, there are no strict criteria for which websites it is essential to browse with HTTPS and which it is permissible to browse without. Often, the risk must be evaluated on a case-by-case basis.

Given that HTTPS provides Confidentiality, Authenticity and Integrity, the new criteria I developed must evaluate in each circumstance what is 'the worst that can happen' if we do not have these guarantees. In many instances, it is all three that are valued. For example, when purchasing online, it is important to ensure our banking details remain confidential and cannot be stolen by third parties. For the same reason, we will value authenticity to ensure that our banking details are being transferred to whom we think.

Integrity is also useful in this case; however, it is also important consider the situations where integrity alone is paramount and confidentiality is not essential. There are many cases when we are communicating with a trusted party but not transmitting confidential information. We must have confidence that the data being transmitted has not been altered or tampered with by a third party. This could perhaps be, when reading the bank account information of a business in order to send them funds or reading a user's cryptocurrency address before conducting a transaction. If an adversary altered the web page to include their bank information or cryptocurreny addresses instead of the legitimate one, the user could inadvertently send funds to the adversary instead. Another more common situation would be reading a news website; we typically do not transmit any confidential information, but a third party tampering could alter a news story to include false information.

Curiously, due to the boundaries in setting up an HTTPS-enabled server, the type of website we are visiting will also have a bearing on whether we require an HTTPS connection. Up until the launch of Let's Encrypt [118] in 2015 [211], obtaining an SSL certificate required some financial expense. Although Let's Encrypt has been available for a number of years, it has not become a complete panacea for the problem of setting up HTTPS. It has also always required some technical expertise to set up an HTTPS site over an HTTP site. The result of this is that the vast majority of larger commercial operations employ the necessary IT staff and are willing to spend the funds necessary to obtain an HTTPS-enabled website, whilst it is not unusual for older, non-commercial websites not to have SSL. This creates a scenario whereby it is surprising and suspicious if a large commercial website does not provide SSL. One would be highly suspicious if tesco.co.uk did not provide SSL, and we were faced with an HTTPS-Only mode warning upon visiting. In contrast, when visiting the personal website of an academic, it would be not unusual to be faced with such a warning.

With these examples in mind, I provide three criteria for requiring an HTTPS connection on the modern web.

1. We trust the information the website is providing to us

2. We are transmitting confidential information

3. A large commercial business or official organisation is running the website

If any of these criteria are met, I propose that the user should not proceed past the HTTPS-Only mode warning.

These criteria are up to interpretation. How much trust is enough to require HTTPS? How large of a business should require HTTPS? This can also vary depending on the country. In the UK, it is expected that all official government or local government websites shall have HTTPS; however, in different countries, this may not be the expectation. It is often the case for example, that official government websites in

China do not provide HTTPS [212]. Given that the study is focused purely on UK participants, it is only necessary to focus on the criteria that make a UK website unsafe for proceeding past the warning.

Users will have different definitions of these criteria, and their concepts of trust, confidentiality and officiality will of course vary considerably. Nevertheless, these criteria can provide a relatively useful first attempt at classifying whether non-HTTPS websites can be considered safe and were used to evaluate the new warning pages that were crafted.

The fact that these criteria are a first attempt can be seen as a limitation of the study. Changes to these criteria would result in different results in the study I conducted in this chapter. Given that no such attempt has been made to create this categorisation before, however, this is a necessary first step. Future research will likely make amendments or additions to these criteria.

5.5 Survey Design and Methodology

In order to establish whether the newly designed warning pages are actually more effective than the current version, the survey is designed to describe certain scenarios to participants and face them with the warning pages. We compare participants' responses against the control page, which is Tor Browser's current HTTPS-Only mode warning page.

To evaluate the effectiveness of these newly designed warning pages, a suitable metric for success is required. In past studies on SSL warnings, the desired outcome is simple: whenever a user receives an SSL error warning, they should not to proceed to the website. Only in a very small set of circumstances should a user continue to a website with this error. The outcome for HTTPS-Only mode warning pages is quite different. Although the majority of popular websites implement HTTPS [114, 115, 116], there is still a significant number of legitimate websites that do not deploy HTTPS and only offer HTTP connections. Therefore, as users browse the web, there is a significant number of sites for which they will receive a warning, and the

optimal action is to proceed through the warning and continue to the website. The success of a warning page, therefore, cannot be measured based on how many users proceed through the warning when displayed.

The measure of success must be dependent on the website being visited. How, then, should it be determined which websites are worth proceeding through and which are not? We propose that users are most at risk of attack on websites where they will enter personal information, in particular financial information. Therefore, websites like banks, social media accounts, and online retailers are all websites where a user should desire HTTPS. Purely informational websites like news websites, train schedules, or restaurant menus are less risky for users, and a lack of HTTPS is not as detrimental as personal information that could be stolen is less likely to be entered by users. It is not the case that SSL is not useful for these websites; indeed, depending on other factors, they may still be quite risky. A news website, for example, if intercepted, could display altered news headlines to deliberately misinform the user or even alter the website completely to scam or defraud the user.

The other factor focused on is website popularity. Typically, very well-known websites will be more likely to have HTTPS[115], lesser-known websites, especially ones which do not represent a commercial business, may not have IT personnel at their disposal to enable HTTPS due to HTTPS deployment being a difficult task [213]. It is for this reason that higher levels of risk should be expected on a website that is very popular yet does not have HTTPS. One can imagine visiting amazon.co.uk and receiving the notification that the website does not have HTTPS would be cause for suspicion. Based on these 2 criteria, 6 web browsing scenarios were devised to present to users during the study. In some of these scenarios it is recommended to proceed past the warning to the websites without HTTPS, and in other scenarios is it recommended to not visit the website. Based on users' responses to each scenario the effectiveness of the new warning pages can be evaluated and compared with the current warning page as a control.

- Visiting your banking website to check your balance

- Buying an item from a large popular online retailer
- Your friend sends you a link to a news website you have never heard of
- You are purchasing a gift for someone on a small independent shopping website
- You are browsing to find the menu for a small local restaurant
- You are visiting a well known reputable news website

These 6 scenarios are presented in random order to participants, and after each scenario is presented, they are shown the warning page that was selected for them. They can either proceed to the website, close the website, or provide a free-text response 'other'. Participants who select other have their responses manually coded to fall into either the proceed or not proceed category. For example, a participant who selects other and states: "I would call my bank to ask for help" would be coded as not proceeding as they are taking other precautions. Alternatively, a participants who states "I would check my antivirus before proceeding" would be coded as proceeding as the antivirus could not detect a man-in-the-middle attack and the participant would likely then proceed. We define the correct answers to these scenarios as follows.

5.5.1 Visiting a Bank: Do not Proceed

Visiting a banking website is a scenario where users should not proceed. An adversary intercepting a connection could acquire a users bank details, or provide the user with other false information. It is also much more likely that banking institutions will have substantial I.T departments and will have ensured that HTTPS is provided. The new criteria clearly indicate that users should not proceed because doing so would require providing confidential information to the websites, the site is trusted, and is a large commercial websites. All three of my new criteria apply and indicate that one should not visit the website.

5.5.2 Large Online Retailer: Do not Proceed

A large online retailer should not be proceeded to for similar reasons to above. An adversary intercepting a users connection could potentially retrieve the users banking or login details for that retailer. Since this is a well known site, it is additionally very unlikely that the website would not implement HTTPS. This scenario is intended to envision a website like apple.com, or ebay.co.uk where it would be highly suspicious if HTTPS was not available.

5.5.3 Unknown News Website: Do Proceed

This scenario presents the user with a news website that they have not heard of before, however they have received a link to it from their friend. The user will likely not have a high amount of trust in the news website as they have not seen it before. The organisation is also not likely a particularly large organisation since the user has not heard of it. For these reasons it is decided that visiting the website is the acceptable choice. The user will not likely be entering any personal information as this is a news website. Additionally, any attacker wishing to provide the user with false information will be thwarted by the users lack of trust in the website. Finally, the user should not assume that because they received the link to the website from a known friend that it is safe.

5.5.4 Purchasing on a Small Independent Website: Do not Proceed

Although in this scenario a less prominent and popular website is being visited, and this factor would normally relieve the necessity of HTTPS, because the user will be making a purchase on the website, the need for HTTPS is heightened. A user must not enter bank card details onto such a website where there is risk of the details being stolen by an adversary.

5.5.5 Menu of a Local Restaurant: Do Proceed

In this scenario, no personal information is being given to the website. The user is also receiving information from the website which could potentially be altered by an adversary. The information being received, however, is likely very low risk

information. It is quite unlikely that an adversary would choose to alter the menu of a user, and even if they did, the user would correct the information once they reached the restaurant. Additionally, many restaurants are small businesses, and thus may not have dedicated I.T specialists, it is likely common that the websites of restaurants do not have HTTPS and so it is not an indication of attack if HTTPS is not available.

5.5.6 Reputable News Website: Do not Proceed

Visiting a well known, reputable news website is likely the most nuanced scenario in this list. Typically, users do not enter personal data into news websites (save for some users who maintain user account there), this negates a large amount of risk from continuing. Because the news website is well known and reputable, it would be highly unusual for HTTPS to not be available and thus it has a much high likelihood of being an indicator of attack. While, the adversary may not be able to steal sensitive data from the user, they can alter news stories on the website and feed the user false information. As the user trusts this website, they are likely to believe information found there and the adversary can thus have a corrupting effect on the user. Proceeding is not recommended.

5.5.7 Tor and the Representative Sample

A key difficulty in this study was dealing with a large representative sample of people who likely would not know anything about Tor or the Tor Browser. Given that this study is specifically concerned with warning pages within Tor browser, it is essential to frame the presented scenarios in this context.

This project attempts to create new warnings for Tor Browser specifically. Ideally, they new pages could be evaluated by users of Tor Browser. It would however, be very difficult to recruit a representative sample of Tor Browser users for this survey. Tor Browser users are often privacy conscious, the Tor project understands this and has not sought to create demographic information on Tor Browser users. Moreover, there does not exist an appropriate survey platform to amass this sample. Tor Browser users are not numerous enough to appear in online survey platforms

like Prolific.com or Mechanical Turk. It would not have been feasible to properly explain Tor to all survey participants within the time allotted, so a scenario was created that would mimic the risks involved in using Tor, except in a way that should have been universally understandable by participants. Reminding ourselves that the key risk of not using HTTPS in Tor is the risk of interference from malicious Tor exit nodes. These exit nodes can alter or manipulate any plain text traffic that flows through them. A HTTPS connection blocks this attack as the exit node cannot simulate the website's SSL certificate. We designed the following scenario to mimic this risk:

"In all the following questions, imagine you are staying at a hotel. Upon check-in the receptionist notifies you that the regular hotel WiFi is broken. They also inform you that another unnamed guest has offered to share their own WiFi connection with other guests, and it should be available under the name: 'Free WiFi'. You connect to this network and continue to browse the web."

This scenario presents the risk of Tor bad exit nodes to participants in a form that should be much more understandable. The result of the participant connecting to the internet via another untrusted hotel guest means that this hotel guest could intercept their traffic if they are not using HTTPS.

The use of this workaround to receive appropriate responses to Tor questions without a sample of Tor users is a significant limitation of this study. It is not guaranteed the data received will sufficiently approximate the experiences of Tor users. That being said, I consider this the best possible approximation that allows a large sample to be used. There is also no obvious reason why this question would not mirror the situation faced by Tor users.

5.5.8 Other Survey Design Decisions

Typically, it would be seen as a limitation of the experiment that the warning pages were not viewed in a real-world browsing scenario. Participants are aware that they are under experimental conditions and may think much more about their actions be-

fore proceeding. Although a more ideal scenario has been achieved by studies like the one by Felt *et al.* [191] it is not a feasible experimental setup in this case. Experiments in those conditions often requires making alterations to the browser itself, which can only be done by the browser manufacturer. Overall, the impact of this limitation is mitigated by the setup. My study compares warnings to each other under the same experimental conditions, and does not compare the results to any real known quantity. The current Tor Browser warning page is compared against the warning pages in the same scenarios, and therefore comparing the results should be equally effected by experimental biases. The study avoids some of the biases that have been introduced in experimental designs [214] due to its like-for-like comparisons. Any biases that users have due to the experimental scenario would be seen across all four warning pages, and thus, their effect on the final result should be broadly negated.

The survey is also less vulnerable to social desirability bias than typical SSL studies. It is not necessarily obvious to participants which is the correct option to select in each scenario. In typical user studies, users can be more likely to adhere to warning pages because their actions are being monitored [215] and they wish to take an answer which will satisfy the experimenters most (particularly if they are not given a separate primary task). Users in general are less familiar with HTTPS-Only modes and thus if they attempt to produce the action they believe to be expected of them, they may make a mistake. If users treat the HTTPS-Only mode warning as any other SSL warning, and do not proceed to any web pages, this will become apparent in the data. This means we can properly acknowledge this effect if it occurs and adjust my conclusions accordingly. It is also the case, as above, that the warning pages are being compared to each other in the same scenario, the result is not an absolute value of the effectiveness of the warning. The result of the study is if any of the warnings show a comparative improvement over the current warning pages, therefore social desirability bias should affect all warning pages equally and not damage the results.

5.6 Results

I used the Prolific.com platform to recruit participants for the study. In total, 1994 participants were recruited and split randomly into 4 groups, one for each warning page. Participants were recruited using Prolific's 'representative sample' option, meaning the participants were broadly representative of the general UK population according to age, sex, and ethnicity.

The vast majority of participants only selected proceed or not proceed answers. A small number selected 'other' in some scenarios and their responses had to be coded manually.

5.6.1 Did Participants Use HTTPS-Only Modes Before?

At the beginning of the survey, participants were asked if they had used HTTPS-Only modes before. The results of this question are displayed in Table 5.2. Approximately 83% of participants reported never having used HTTPS-Only modes before which comports with the features relatively recent deployment.

Table 5.2: Participants reporting on whether they used HTTPS-Only modes before

	Count	Percentage
Yes	336	16.9
No	1654	82.9
Declined to answer	4	0.2

5.6.2 Evaluation of New Warning Pages

To evaluate the effectiveness of these new warning pages, I conducted significance tests on each of the 3 warning pages. In these tests, I compared whether users proceeded past the warning page against other users' actions on the control warning. Assessing users' actions in this way provides a good insight into the warning pages' effect, but it is also true that the warning page could affect users in other ways. One example of this would be changing how users behave when they proceed to the website. Therefore, this method evaluates most, but not all, of the effects of the warning pages. For each page, the total number of 'correct' actions was summed and com-

pared to the total number of 'incorrect' actions as decided by the new criteria. When participants provided their answer as to if they would proceed or not, they could either 'proceed', 'not proceed', or, 'other'. These values are coded depending on the scenario coding as discussed in Section 5.5. Theses codings are included also in Table 5.3 for convenience. A chi squared test was conducted for each page compared to the control. The Holm-Bonferroni method was used to prevent multiple comparisons errors. The results of this testing can be seen in Table 5.4. The popularity page was found to have statistically significantly better outcome than the control when all scenarios are grouped. The ϕ coefficient of 0.0255 indicates a small effect size; the improvement made by this warning page is minor.

Table 5.3: Safety coding for each scenario

Scenario	Recommended action
Bank	Do not proceed
Large Retailer	Do not proceed
Small Retailer	Do not proceed
Unknown News	Do proceed
Restaurant Menu	Do proceed
Reputable News	Do not proceed

Table 5.4: The effect on participants actions for each new warning page. Scenarios are grouped to show the overall effect. Only the popularity warning page shows significant change.

Warning	Correct	Incorrect	Total	Test Statistic	p-value	Corrected α	ϕ coefficient
Control	1989	1011	3000				
Popularity	2037	927	2964	3.8862	**0.0487***	0.05	0.0255
Context	1990	1016	3006	0.0029	0.9570	0.025	0.0007
Tor	1998	996	2994	0.1077	0.7428	0.0167	0.0042

* Statistically significant

5.6.3 Warning Page Effectiveness by Scenario

The warning pages were next assessed by each scenario. This resulted in 18 additional statistical comparisons, giving a total of 21 tests. It was exceptionally im-

portant to use a corrected α level to avoid erroneous conclusions being drawn due to multiple comparisons false discovery errors. The Holm–Bonferoni method was again used to give corrected levels. These results can be seen in Table 5.5. The reputable news scenario was the only one to feature statistically significant changes to the participant response rates. The popularity warning had a statistically significant improved score, and the context warning had a statistically significant dis-improved score. Both pages had a similar ϕ value of approximately 0.1, indicating a minor effect size.

The context warning page produces two other results which were *almost* significant. The large retailer scenario saw a dis-improvement with an equivalent p-value of 0.16 and the restaurant menu scenario with an equivalent p-value of 0.14. While these are not significant results, they may indicate that the context warning is worthy of further investigation.

A more digestible form of the user actions data can also be seen in Figure 5.4 which shows the breakdown of correct answers by scenario and warning page. It can clearly be observed that there is an overwhelming discrepancy between scenarios which were judged to warrant proceeding past the warning and those where not continuing to the website was desired. It is clear that users are not proceeding to website even when they are websites where this was deemed acceptable. This major difference exists across all four warning pages tested. The first three scenarios where proceeding is not recommended show very little variation depending on warning page, changes of 1-3% can be seen which were not large enough to show effect in statistical testing. As for the next three scenarios, the unknown news scenario only shows difference on the Context warning where a 5% improvement is seen. On the restaurant menu scenario the context warning also achieves an 8% improvement. Again, neither of these were statistically significant and perhaps only indicate an area for future research to be done. The real major differences can be seen on the reputable news website scenario, this is the one scenario where statistically significant changes took place. The Popularity warning achieved a 10% improvement in

Table 5.5: The effect on participants actions for each new warning page and each specific scenario, showing p-value and Holm–Bonferroni adjusted alpha values. Only two scenario-warning pairs are significant, both in the reputable news website scenario.

Warning	Scenario	Correct	Incorrect	Total	Test Statistic	p-value	Corrected α	ϕ coefficient
Control	Bank	488	12	500				
Popularity	Bank	488	6	494	1.3537	0.2446	0.0125	0.0369
Context	Bank	491	10	501	0.0485	0.8256	0.01	0.0070
Tor	Bank	489	10	499	0.0445	0.8330	0.0083	0.0067
Control	Large Retailer	459	41	500				
Popularity	Large Retailer	465	29	494	1.7195	0.1898	0.0071	0.0416
Context	Large Retailer	437	64	501	5.1007	0.0239	0.0063	0.0714
Tor	Large Retailer	456	43	499	0.0153	0.9016	0.0056	0.0039
Control	Small retailer	60	440	500				
Popularity	Small retailer	58	436	494	0.0008	0.9775	0.005	0.0009
Context	Small retailer	84	417	501	4.2374	0.0395	0.0045	0.0651
Tor	Small retailer	58	441	499	0.0075	0.9311	0.0042	0.0027
Control	Unknown News	465	35	500				
Popularity	Unknown News	451	43	494	0.7765	0.3782	0.0038	0.0279
Context	Unknown News	471	30	501	0.2719	0.6021	0.0036	0.0165
Tor	Unknown News	474	25	499	1.4172	0.2339	0.0033	0.0377
Control	Menu	175	325	500				
Popularity	Menu	190	304	494	1.1367	0.2863	0.0031	0.0338
Context	Menu	217	284	501	6.9145	0.0085	0.0029	0.0831
Tor	Menu	171	328	499	0.0311	0.8599	0.0028	0.0056
Control	Reputable news	342	158	500				
Popularity	Reputable news	385	109	494	11.0198	**0.0009***	0.0026	0.1053
Context	Reputable news	290	211	501	11.4424	**0.0007***	0.0025	0.1069
Tor	Reputable news	350	149	499	0.2783	0.5978	0.0024	0.0167

* Statistically significant

action by users, and the Context warning had a 10% reduced correct actions both of which were statistically significant changes. The Tor warning only improved 2% in this scenario and was not statistically significant.

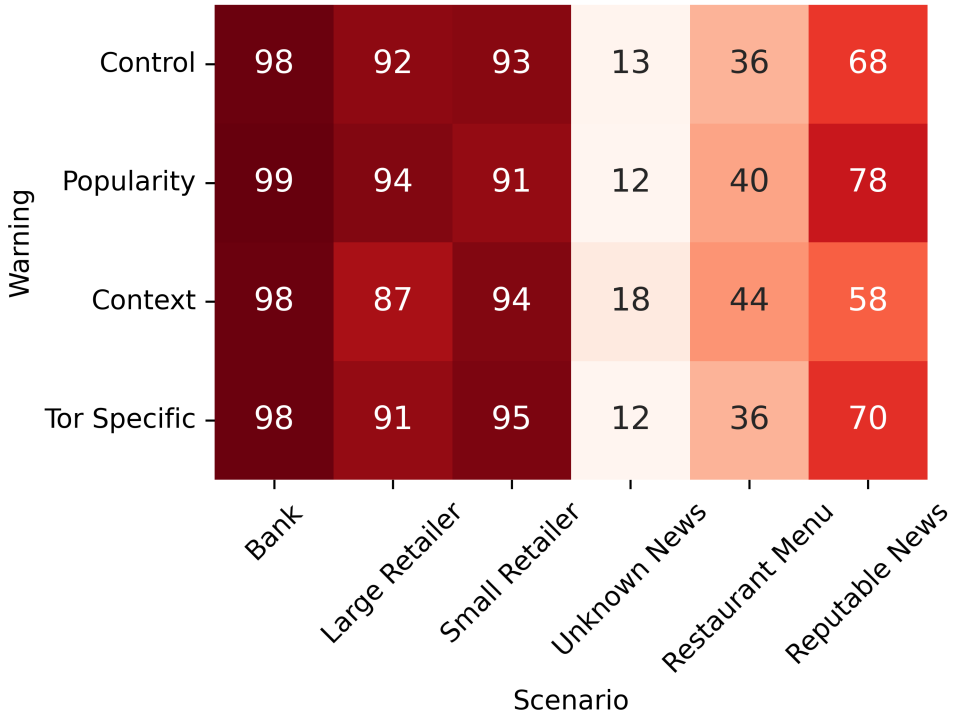


Figure 5.4: Percentage of correct actions by scenario. The numbers shown here are not necessarily statistically significant. Please see Table 5.5 to see which pages and scenarios demonstrated statistically significant difference from the control task.

5.6.4 Warning Page Dwell Time

There was no significant difference in the time users spent on the warning page before proceeding. The Two-sided Mann-Whitney U test was used as the samples are independent and it is not certain that the distribution of the data is normal. The result is displayed in Table 5.6

5.6.5 Demographics

As a result of the representative sample provided by the Prolific platform the sample closely mirrors the demographics of the general UK population. Demographic information is provided in Table 5.7. Minors were not recruited for ethical reasons.

Table 5.6: Dwell time results: there was no significant difference in the time spent on the warning pages compared with the control warning. (Correction was not used as none of the results were close to being significant.)

Warning Page	Time on Page (s)	p-value
Control	174	
Popularity	192	0.14
Context	167	0.69
Tor	179	0.95

Table 5.7: Demographic information of the participants

Characteristic	Count	Percentage
Male	970	48.6%
Female	1024	51.4%
White	1742	87.4%
Mixed	29	1.5%
Other	18	0.9%
Black	62	3.1%
Asian	143	7.2%
18-29	405	20.3%
30-39	363	18.2%
40-49	343	17.2%
50-64	621	31.2%
65+	258	13.0%

5.7 Discussion

5.7.1 Improvement in the Popularity Warning

The Popularity warning was the only new warning page to show significant improvement overall. This means that, by the criteria given in this study, this warning page is superior to the control warning. When considering the scenario specific data, the Popularity warning only showed statistically significant improvement on the Reputable News website scenario. A likely conclusion of this result is that the Reputable News websites scenario is the scenario that is most un-intuitive for users, and that the popularity warning is the warning which addresses this scenario most effectively. The particular effectiveness of one warning for a specific scenario

potentially motivates to the inclusion of multiple design criteria in a more comprehensive warning page. Warning about site popularity could be combined with other warnings that are more effective at other scenarios.

The relative un-intuitiveness of the Reputable News websites scenario may also indicate that improvements will be harder to generate in the more obvious scenarios. This could indicate the need for much larger study samples to detect the smaller improvements that will be generated by improved warning pages.

5.7.2 Lack of Improvement of Context and Tor Based Warnings

The Context warning did not provide statistically significant results overall. The specific scenario analysis showed significantly dis-improved results for the reputable news website scenario. It can also be seen in Figure 5.5 that the Context warning is close to providing significant result for the Restaurant Menu scenario also. While the results were not statistically significant, this still indicates that a statistical effect could be present and thus perhaps an effect could be discovered if this style of warning page was pursued in further research. The context warning as a whole suffered from the dis-improved reputable news results and so it is not likely to be a complete improvement on its own, however, some of the design from the context warning could potentially be useful in future improved warning pages.

Failure to show the effectiveness of the Tor driven warning could also be caused by the experimental design. The scenario developed to mimic the risks faced by Tor users was innovative, but it cannot be said to be a perfect analogue to a genuine user of Tor browser. The scenario developed involved users connecting to the internet via a shared connection from an anonymous hotel guest. The fact that users were connecting through this unknown connection could allow the connection owner to eavesdrop or alter the content of their unencrypted connections. This mirrors the abilities of malicious Tor exit nodes. The first issue with this scenario is that the participants of the study, representing a general UK population, may not have accurate enough mental models of web browsing and HTTPS to understand the implications

of the scenario presented. This may contrast with Tor Browser users, who may have more accurate views of how the information presented by the warning page applies to their use of Tor. Although Tor Browser aims to be usable by individuals from any kind of technical background, it is likely that users are more frequently from a technical background. Additionally, Tor Browser users, regardless of their technical expertise, necessarily chose to use Tor, and thus, they are more likely to be concerned about security and privacy. Therefore, their mindset when receiving a warning may be quite different and thus may not be analogous to the study situation.

In summary, while these two warnings did not show significant improvement, there is still potential for a Tor specific warning to be created that advises users of specific Tor threats. The Context warning could also be studied further as there may be potential in this design theme for persuading users to proceed to websites which are not risky. Analysis of any such warning, however, may be difficult and could require more innovative experimental design.

5.7.3 Users are Hesitant to Proceed to Less Risky Websites

The data from this experiment clearly shows that participants are more conservative with HTTPS-Only mode warnings than is necessary. Participants did not proceed to unsafe coded websites to a very high degree ($\approx 90\%$) however many participants did not proceed past the warning to websites which were coded safe ($\approx 27\%$). Some of this difference could be attributed to experimental biases, in particular, as this was not a study of users in their real browsing environment, and participants may have been biased by the focus the experiment placed on the warning pages.

Participants in studies of this type are also usually considered to be under the influence of the social desirability bias, where participants typically answer questions in a way that they feel is more socially acceptable or that will please the researchers more. If participants were under this influence, they would be more inclined to answer what they view as the correct answer rather than the answer which they believe they would give in a real-world browsing scenario. As a result, social desirability bias would bias participants towards giving ‘more correct’ answers, which

in this case means proceeding past the warning page in 'safe' coded scenarios. This negates social desirability bias as a factor for explaining the discrepancy.

Overall, suppose experimental factors explain some part of this discrepancy. In that case, it is unlikely to explain more than a fraction of it. So, it is concluded that the main area to be addressed in further improving warning pages is convincing users that it is acceptable to continue past the warning in these safer scenarios.

5.7.4 The Reputable News Website Scenario

Out of the four scenarios where users should not proceed past the warning page three of them showed very high compliance rates for all warning pages (98%,91% and 93% on average). The reputable news website scenario was the only one to achieve distinctly lower rates of compliance (69% on average). The scenario is distinct from the others in that it is the only scenario where users' personal information is not likely to be required to use the website. Users potentially considered the lack of risk to personal information to be sufficient to proceed past the warning and view the website, despite the scenarios crafted here deciding that proceeding was unwise. It is likely true that users face much less danger from websites in this scenario, and so lower compliance from users does not need to be considered a disaster, it is however reasonable to attempt to increase compliance further with improved warning pages as is shown by the Popularity warning crafted in this study.

5.7.5 Did the Warning Pages Promote Proceeding to Safe non-HTTPS Websites?

All three new warning pages contained lines that specifically attempted to inform users of situations where they were not at risk.

- "If this is not a popular, professional website, you are less at risk." (Popularity)

- "If you read this site without entering personal details, you are less at risk" (Context)

- "If you fully trust your internet connection and any intermediaries, you are less at risk" (Tor)

This is in comparison to the control, current Firefox warning, which states:

- "Most likely, the web site simply does not support HTTPS."

One potential theory for the failure of the warning pages to convince users to proceed to safe coded websites is that the statements in support of safe non-HTTPS websites were not well understood. Potentially only the context one was readily understandable by users. Participants perhaps easily understand the concept of 'entering personal details' on a website. Conversely, they may not understand how to assess whether a website is a 'popular, professional website'. Finally, it is possible that participants were confused by the interplay between the warning about trusting their internet connection and the hotel WiFi scenario. These theories were, of course, not tested and can only be taken into future research as a hypothesis.

5.7.6 Lack of Comprehension

The data also shows evidence that users do not comprehend the warnings properly. The above argument showed that users do not know when it is safe to proceed, however, a more general interpretation of that point is that users show poor comprehension of the warning and their attitude is to simply not proceed past any type of warning page. This conclusion is uncertain, and so more investigation is needed to confirm this hypothesis.

5.7.7 Actions for Web Browser Vendors

From the results of this research, web browser vendors, particularly the Tor Project, who maintain Tor Browser, could directly implement the new Popularity warning page I developed in this project. Further to this, the other new warning pages created in this project displayed some promise in increasing users' compliance. Browser vendors can study these warning pages further to guide the design of their HTTPS-Only mode warning pages. These insights are not only applicable to Tor Browser. The other non-Tor-focused warning pages can provide insight and inspiration for all

other browser vendors.

It is also in the interest of browser vendors to consider the future work outlined in Section 5.9. If implemented, this future work could significantly enhance their products' security and user experience. Funding and engaging with researchers to complete this work would likely be highly beneficial for their web browsers, paving the way for a more secure and user-friendly browsing experience.

5.8 Limitations

5.8.1 Unusual Tor Network Imitation Scenario

Participants may not fully grasp the implications of using a WiFi connection offered by the unnamed guest. As the warning mode pages are intended for Tor browser, ideally, the survey would recruit participants who are users of Tor browser; however, it would be effectively impossible to recruit a representative sample of Tor Browser users due to their inherent desire for privacy. This also means the warning page specifically designed for Tor had to be redesigned, and as such, the tested design is not the same as the design that could actually be proposed for Tor Browser to implement. This degrades the accuracy of the experiment for this warning, although it still provides valuable information on an overall design concept which could be implemented.

5.8.2 Results are Based on New, Untested Criteria

The success of each newly designed warning page is evaluated on my own designed criteria. It was determined whether a user should or should not proceed to a variety of website types based on these three new criteria. There was no robust criteria available on when a user should proceed and so this decision had to be decided using a first approximation which could be challenged or improved in the future.

5.8.3 Survey Gains a Limited Insight into how the Warning Effects User's Behaviour on Proceeding

It is possible that the warning pages change users behaviour when proceeding to websites. This could happen conscious or subconsciously, and in either case this behaviour is not properly studied by this survey. In situations where I state the user should not proceed to a website, the warning may alter a users behaviour such that will no longer trust the content on the website and they will ensure not to enter any personal details. This would remove the risk from the website even though the action was still coded by unsafe in the survey.

5.8.4 Location Specific

The scenarios created for the study are location specific. As mentioned previously, some heuristics used by users may be more or less useful depending on the country. HTTPS adoption varies in different countries and the advice used in the new warning pages could produce different results in different jurisdictions.

5.9 Future work

5.9.1 Testing the Comprehension of HTTPS-Only Mode Warnings

As Felt *et al.* could not show in their experiment that their SSL warnings were not well comprehended by users, it is even more necessary to check if users understand HTTPS-Only modes, not just whether their adherence is good. An evaluation of comprehension would ensure that the results presented here are more generally applicable to other web browsing scenarios, and not just the 6 tested here.

5.9.2 Do Users Need to Proceed to Insecure Websites?

In the previous chapter, some survey participants wanted to have warnings direct users never to continue to non-HTTPS websites. In this chapter, the study showed that regardless of whether this was the intended warning design, users did not continue to non-HTTPS websites at a high rate, even in scenarios where it should be safe. Further debate and discussion on the legitimacy of this outlook is necessary.

5.9.3 Users Need to Understand when to Proceed

If users are deemed to require the ability to still visit websites that do not deploy HTTPS, then it is clear that further design changes are necessary to enable this. Although this study was successful in improving the rates of users continuing to safe non-HTTPS websites, the click-through rate was still dramatically lower than the rate of users not proceeding to potentially dangerous websites. Similar research must be continued to raise this click-through rate and ensure that users are not effectively blocked from visiting safe non-HTTPS websites.

5.9.4 Combining these Design Themes Could be Effective

The Context design theme did not produce significantly improved results across any scenario, however, some of the scenario were almost significant. It may be the case that this design theme could produce significant results if further improvements were made. The circumstance where the Context page was almost significant (the restaurant menu) was also a scenario where the Popularity page did not achieve improvement. Similarly, the reputable news scenario resulted in significantly improved result for the popularity warning, but dis-improved result for the Context warning. This could indicate that if both of these design themes were combined, they would be more effective. Further work could be done in designing a warning page that combined the themes and tested the result of this new warning page.

5.10 Summary

This chapter discussed how 3 new warning pages were designed for HTTPS-Only modes in Tor Browser. Discussion continued to the new criteria that were developed for use in evaluating these new warning pages. I provided an outline of the experiment that was performed where participants gave their reactions to the warning pages under different web browsing scenarios. A warning page was considered successful if participants did not proceed to unsafe-coded websites that did not support HTTPS but did proceed to safe-coded websites that did not support HTTPS. The study showed that one of the new warning pages statistically significantly improved users responses. Several other conclusions were also apparent. Most prominently,

it could be seen that participants did not proceed to unsafe websites at a high rate but also did not proceed to safe websites nearly as much.

The work provides numerous questions for further study in how users do and should approach HTTPS-Only mode warning pages.

Chapter 6

CoStrictTor: Bringing HSTS to Tor browser

6.1 Introduction

HSTS (HTTP Strict Transport Security) is an HTTP security feature that forces secure connections on websites that have previously provided a HSTS flag to the user's browser. HSTS was previously discussed as one of the primary methods of avoiding SSL Stripping attacks due to the fact that HSTS forces HTTPS connections and will not allow them to be degraded to insecure HTTP connections.

HSTS therefore seems to be an excellent solution to the outsized problem of SSL Stripping by bad Tor exit nodes. Currently, however,, the Tor Browser does not support HSTS due to the tracking potential of HSTS. Malicious websites could set arbitrary HSTS flags by directing the browser to load resources from various domains or subdomains, creating a persistent fingerprint allowing multiple visits by the same user to be linked [216]. Disabling HSTS removes the security protection this feature offers and increases the risk that malicious Tor exit nodes could perform a man-in-the-middle attack, and in particular, an SSL Stripping attack.

This final project in this thesis was completed as part of an equal collaboration with my colleague Dan Ristea. Together, we constructed a protocol which enables

users of Tor Browser to effectively crowdsource their HSTS flag status. Users share HSTS flags they receive from websites into a centralised location where they can be distributed to other users for querying. This eliminates the potential of HSTS tracking as described in [216] by ensuring that any malicious HSTS flag will be shared among all of the users of the protocol.

To inform our design decisions, we survey the current state of HSTS deployment, including the distribution of expiry times. We use the Majestic Millions index of the most popular websites based on incoming links.

One immediate consideration of a protocol like this is privacy; if users share data on websites they visit, it is essential to introduce a privacy mechanism such that users do not automatically reveal which websites any one user has been browsing. To this end, the core of our protocol involves privacy-preserving data sharing, i.e., a protocol to ensure data can be reliably reported from many clients to a server without leaking the data of any one individual to the server. We have selected the RAPPOR protocol [217] to provide local differential privacy for user submissions. RAPPOR has the necessary flexibility to enable us to use broad concepts from the protocol while also allowing us to alter and build on top of it to adapt it for this specific application. This includes a double Bloom filter construction, which reduces false positives in our protocol and improves its resilience to Sybil attacks. The protocol design prevents it from degrading the user experience, even in worst-case scenarios.

We perform a simulation of the protocol, which allowed us to select the best parameters for the protocol and demonstrated that it can efficiently model up to 150 000 websites.

The CoStricTor protocol provides additional information to the user in a potential man-in-the-middle downgrade attack. It replaces the standard HTTPS-only warning with a variant of the HSTS warning, informing the user that continuing to the page carries a higher risk than the more common HTTPS-only warning. The protocol does not increase the total number of warnings a user will see. As well as helping

users directly, the protocol assists with detecting potential malicious exit nodes in the Tor network, which may help prevent future attacks.

Section 6.2 provides relevant background information on Tor, Tor Browser and HSTS which are necessary for understanding our protocol and why it is useful. We describe our protocol in section 6.4, in which we provide a discussion on various building blocks which make up our protocol before describing the detailed processes of the protocol's operation. The simulation which was used to evaluate the protocol is discussed in section 6.5, and we provide the results which both prove the usefulness of the protocol and describe the appropriate parameters which are needed to operate an implementation of the CoStricTor protocol. We provide a discussion on the usefulness of the protocol as well as its limitations in section 6.6 Related work and conclusions are given in sections 6.7 and 6.8.

6.2 Background

6.2.1 HSTS Supercookies

An unfortunate side-effect of HSTS is its potential as a tracking system. By saving the HSTS status of multiple subdomains, a browser can be manipulated to store a unique identifier which will implicitly be sent to the website on subsequent connections. This may be used as a form of web tracking [3, Sec. 14.9]. Such tracking techniques, which eschew the normal controls for client-side data retention, are termed "supercookies" [216].

To create an HSTS supercookie, a website can include dummy resources from multiple subdomains. On a user's first visit, all connections to subdomains will be made over HTTP, identifying the user as a new user. The server can set a unique combination of HSTS flags for the subdomains in its responses to the user. Therefore, each subdomain's HSTS entry can store one bit of identifying information. On subsequent visits, the pattern of HTTPS and HTTP connections to subdomains based on the unique combination of HSTS flags stored in their browser can identify the user.

Tracking through this method becomes even more effective if it is performed by

actors whose resources are loaded on multiple websites, such as ad networks or content distribution networks. In this case, not only can a user be identified on one site, they can also be tracked across the web. To prevent tracking through shared resources, including HSTS, Firefox introduced cache partitioning, which only allows access to local resources to the top-level domain that set them. This prevents cross-domain tracking through HSTS but does not prevent tracking over time on a single domain [218].

Apple has also introduced protection mechanisms through its WebKit framework, which significantly reduces the scope of HSTS tracking [219].

In 2015, Zhu [220, 221] proposed another tracking method using HSTS allowing a malicious website to check if the visitor had visited known HSTS sites in the past. This exploit is possible in all web browser, but was most effective in Google Chrome until it was partially fixed [222]

6.2.2 Tor Consensus

The configuration of the Tor network is decided through consensus by a set of *directory authorities* whose IPs are hard-coded into Tor clients [223]. When a Tor client initiates its first session it will connect to a directory authority to download the network configuration but subsequent updates will be downloaded from directory mirrors. The consensus is approximately 2.8 MB or 580 kB compressed. Once downloaded, consensus data is cached while it is valid and persists between sessions. Whenever a client initiates a new connection, it uses the configuration to pick a new set of relays to route their traffic.

Some nodes in Tor are designated *guard nodes* which are considered more trustworthy than other nodes. These nodes are used as the entry, or first node in a users connection to Tor. Tor clients select a single guard node to be used for a period of time and all connections made will begin with that guard for the duration.

The Tor network has been subject to multiple attacks which aimed to compromise the encryption of outgoing connections. Due to its position, the Tor exit node

can perform man-in-the-middle attacks when forwarding packets to the destination server. SSL stripping attacks, in which a man-in-the-middle prevents the upgrade from HTTP to HTTPS have been used to deanonymise communications [31] and to rewrite cryptocurrency transactions [33]. Due to the outsizes impact of bad exit nodes, Tor provides a mechanism to report potentially suspicious exit nodes [224].

6.2.3 Tor Browser Deletes HSTS Flags

As discussed, Tor Browser is designed to protect users' privacy. By default, it hard-codes identifying information the browser exposes to servers, which creates a standard fingerprint shared by all Tor browser users. To further impede tracking attempts, it deletes local data between Tor sessions, including HSTS flags. Although this behaviour prevents the use of HSTS supercookies to track Tor Browser users across sessions [38], it negates the intended role of storing HSTS flags long-term. Tor Browser does include the HSTS preload list and uses this to enable HSTS in a limited fashion [225].

As of version 11.5, Tor Browser defaults to HTTPS and displays a warning page before allowing users to continue over an insecure HTTP connection [226].

6.2.4 Differential Privacy

Differential privacy is a definition of privacy first introduced by Dwork *et al.* in 2006 [227] and it has been widely adopted in privacy-preserving data collection and manipulation.

Differential privacy is a property of a mechanism for aggregate data retrieval. It protects any individual's privacy by making any observed output of a mechanism operating on the data almost as likely as the output of the mechanism if the individual's data was not present. More formally, mechanism M is said to be ε-differentially private if the following inequality holds for any two neighbouring databases D and $\hat{D}$ that differ in one item and for all subsets S of the possible outputs of M, i.e., $S \subseteq \text{Range}(M)$:

$$\Pr[M(D) \in S] \leq e^{\varepsilon} \cdot \Pr[M(\hat{D}) \in S]$$

The privacy parameter ε, also termed the privacy budget of the mechanism, provides a quantifiable measure for the relative privacy loss.

An important property of differential privacy is *immunity to post-processing*. Differentially private data can be released in full without any further degradation of privacy, regardless of any auxiliary data or computational resources an adversary might have.

Differential privacy mechanisms are implemented by introducing random perturbations, such as adding randomly sampled noise, to create uncertainty over the data which produced an observed result. In the centralised setting, the data will be aggregated first and then have a DP mechanism applied to produce the desired results, thus it requires a trusted collector; in the local setting, data is perturbed to ensure differential privacy before being sent to a collector, thus dispensing with the need for a trusted collector.

A mechanism, M, has local differential privacy if for any observed output $s \in \text{Range}(M)$, and for any two inputs x and $\hat{x}$:

$$\Pr[M(x_1) = s] \leq e^{\varepsilon} \cdot \Pr[M(\hat{x}) = s]$$

The idea of perturbing responses before they are collected predates differential privacy by decades [228]. Randomised response was introduced to avoid the bias inherent in research into sensitive topics. To answer a question using randomised response, an individual will secretly flip a coin and report the truth on heads or flip another coin to decide their response. Each response thus has the plausible deniability of being the product of a random process, but the real count of positive responses can be approximated based on the observations. Generalising, in the local setting,

differential privacy ensures that even if the aggregator and all participants bar one collude, they cannot compromise the value submitted by an honest participant. This comes at a cost to utility as the overall noise introduced by local differential privacy is higher than in the centralised setting.

6.2.5 RAPPOR

RAPPOR [217] is a protocol for private aggregate data collection introduced by Erlingsson et al. in 2014. It generalises randomised response to encode arbitrary data through the use of Bloom filters. The protocol was integrated into the Google Chrome browser [229] but was subsequently removed.

RAPPOR has two phases. Firstly, a reporting phase in which user data is encoded into a Bloom filter, which is randomised for local differential privacy, and sent to the server where it is aggregated. Secondly, a decoding phase in which the aggregated reports are processed to obtain the relevant statistics. This process provides local differential privacy to the individual user reporting the data, to the extent allowed by the protocol's privacy parameters.

6.2.5.1 Reporting

RAPPOR aggregates reports which encode data over a specified time period. A client will report a particular value so it can be tallied without revealing it to the server. To send a value to the aggregation server, it is inserted into an empty Bloom filter B of size k.

RAPPOR provides privacy by randomising the Bloom filter locally in two steps. It first generates a *permanent randomised response* (PRR) B', which will be used for all further reporting of the same data. See [217] for details on how PRR is computed using the privacy parameter f.

Before reporting to the server, the value is again perturbed in an *instantaneous randomised response* (IRR) S. The IRR is computed on every reporting on the value by initialising S with all 0 bits and then setting the value of individual bits S_i as follows:

$$P(S_i = 1) = \begin{cases} \text{q}, & \text{if } B'_i = 1. \\ \text{p}, & \text{if } B'_i = 0. \end{cases} \tag{6.1}$$

where the probabilities p and q are parameters of the RAPPOR protocol. The client will finally send the IRR S to the server, where the set bits are aggregated with all other responses.

This system preserves the deniability of each report from randomised response. It introduces longitudinal privacy so that even over multiple reports of the same value, the reports cannot be correlated, and an averaging attack cannot be performed to infer the underlying value. Our protocol omits the PRR step, as described in section 6.4.3.1.

6.2.5.2 Decoding

Upon receiving reports from clients, the server tallies these reports to get the overall number of bits set, accumulating the individual instances of Bloom filters into one counting Bloom filter , which we denote as c. The true counts of these bits in the filter , t_i, can then be estimated using the parameters which encoded the reports, f, q, p as well as N the number of total reports (see the appendix of [217] for more details). From the array of estimated counts t, RAPPOR produces a best-fit distribution over a dictionary of known values using regression. Our protocol omits the regression step, as described in Section 6.4.3.4.

6.2.5.3 Hash Functions

RAPPOR uses multiple hash functions to insert the data into Bloom filters. A higher number of hash functions h reduces the chance that two distinct elements map to exactly the same bits, reducing false positives – but decreases the maximum number of items that can be reliably stored before false positives start increasing again. The same set of h hash functions is used to check membership of the set. Our protocol does not deviate from RAPPOR in how it uses hash functions. As we expect many more insertions compared to the size of the filter, the optimal value of h is always 1,

therefore, we omit h from further discussion. For additional information on h, see [217].

6.2.5.4 Cohorts

One of the key issues with bloom filter-supported protocols are false positives, bloom filters will never produce false negatives, but as more items are added to the filter, false positives become more and more likely. To combat this, the authors introduced the concept of cohorts to the protocol. Whenever a user joins the protocol, they are assigned a cohort number, from $0..m$ where m is chosen beforehand as the number of cohorts being used. Each cohort uses a different hash function to insert into its bloom filter, and each user reports their cohort number when reporting data to the server. This amounts to running m different instances of the protocol at once, except with different hash functions making false positives which appear in one cohort likely not to appear in a different one. A larger number of cohorts reduces the likelihood of false positives. This however is at the cost of reduced 'signal strength', i.e. if the protocol divides all users into these cohorts, the presence of any one string in the set will be harder to detect which is why we do not employ cohorts in our protocol.

6.2.5.5 Privacy

Reporting data through RAPPOR, as used in CoStricTor, provides local differential privacy with

$$\varepsilon = \log(\frac{q(1-p)}{p(1-q)})$$

Recall that p is the probability of any 0 bit being falsely reported as a 1, and that q is the probability that any 1 bit is correctly reported as a 1. Using higher values of p and lower values of q introduces more noise to reports but reduces accuracy. For additional information on ε, see the RAPPOR publication [217].

6.3 Data Collection

6.3.1 Web Scraping

To properly evaluate our protocol, we required accurate and up-to-date data on the deployment of HSTS.

Some surveys of HSTS adoption exist in the literature. One anonymous HSTS study [230] has its latest results in 2019, however comes with little information on the methodology of data gathering, and the more rigorous work by Garron *et al*. [231] which although features extensive discussion and reasoning, is only from 2013, meaning that the data is very out of date at this point. W3Techs provide a dataset [232] which is very up to date, publishing monthly data for several years, however, once again no methodology information is provided. Work by de los Santos [233, 234] also provides data on this with valuable information of the presence of "include subdomains" directives, however once again, this data is too old for our uses.

As a result, we conducted web scanning of a sample of websites to obtain an up to date and appropriate picture of the proportion of websites deploying HSTS. As our further work relates to Tor browser, we also take a specific interest in the level of HSTS deployment in sites which are popular in countries which have high levels of Tor browser usage.

In particular, we aimed to answer two specific questions:

- Of the most popular websites, what percentage deploy HSTS?
- Of these, what expiry times are most commonly used?

For surveying the most popular websites, we used the Majestic Million list of websites [65]. This is a freely available index, based on the number of other domains referring to the host. We then deployed the Zgrab utility [235] to perform a minimal scan: a single HTTP request for the root of each website, producing minimal load on the services. The only data stored was HSTS headers. As a one-off scan

with a single request, an opt-out mechanism would not have had any effect. This would have been offered if we planned any further scans. However, we omitted to provide information to web administrators about the purpose of our scan, deviating from best practices. We deemed there to be no chance of overwhelming the network infrastructure of these websites with our scan due to their popularity.

The result can be seen in Figure 6.1. We observe that approximately 15% of the top 1 million websites have HSTS enabled. For HSTS to be effective, the expiry time must be long enough for users to return to the website. A proportion of websites surveyed have expiry times that are too short for this to occur. This can be partially explained by some websites being in the process of rolling out HSTS. Best practices recommend expiry times start small – a few minutes – to avoid bad configurations being stored by browsers, then increased gradually to the recommended 2 years [63]. Some unusual configurations can likely be explained by administrator error, after the launch of HSTS, Kranch and Bonneau [236] studied websites with HSTS enabled and found many websites had misconfigured HSTS directives which did not comply with the standard. It is possible that difficulty in configuration from administrators has continued to some extent. Further to this, 9322 of the sites surveyed do have HSTS enabled but with an expiry time of 0, thereby entirely disabling HSTS. The vast majority of HSTS enabled websites have an expiry time of either 6 months, 1 year or 2 years.

This data, as we will discuss later, is critical in properly evaluating our protocol.

6.3.1.1 Country Specific Sample

One of the goals of this work is to find data which is directly applicable to the usage of Tor Browser. we have directly focused on gathering data from countries where Tor usage is highest. The importance of gathering location specific data on both HTTPS and HSTS adoption is highlighted in [116] which notes the large discrepancies in HTTPS adoption across the world. The URLs used were taken from ‘Alexa top sites’, and were the ‘top 100’ most popular sites in 7 of the most popular Tor browser countries as stated in Tor’s metrics [237].

Table 6.1: Frequency of HSTS expiry times

Expiry time (seconds)	Equivalent to	Occurrences
31536000	1 year	70014
63072000	2 years	19248
15768000	6 months	10087
0	HSTS disabled	9322
15552000	180 days	7654
300	5 minutes	7399
7889238	91 days	5295
120	2 minutes	2861
2592000	30 days	2840
15724800	182 days	2614
86400	1 day	1584
10886400	126 days	1535
16000000	185 days	1451
16070400	186 days	1420
43200	12 hours	1214
31557600	1 year, 6 days	1168
604800	7 days	1067
600	10 minutes	835
3600	1 hour	803
31556926	1 year, 5 days	722
31622400	1 year, 1 day	665

We surveyed the top 100 websites in the following countries:

- Belarus
- China
- Indonesia
- Iran
- Russia
- Ukraine
- USA

The results from this survey can be seen in Table 6.2. It is generally clear that there is a difference between HSTS deployment based on country. The highest deployment rate is 47% in the USA, and the lowest is 10% in China. These results are merely a snapshot into only the 100 most popular websites. The results indicate

that security deployment can vary by country, and when designing our protocol we should consider not only on the deployment of HSTS for the most popular sites worldwide, but how it may differ in different regions where Tor is popular.

Table 6.2: HSTS deployment by country for most popular 100 websites

Country	HSTS	Non-HSTS
Belarus	36	64
China	10	90
Indonesia	38	62
Iran	23	77
Russia	29	71
Ukraine	39	61
USA	47	53

As an aside, the recent closure of the Alexa top sites platform has halted any advancement of this data [238]. Data from more sites, i.e. more than the top 100 sites per country cannot be obtained, and more countries cannot be included in the data. At this time it appears no other organisation is publishing country specific data on website popularity.

6.4 Protocol

As long as there are a significant number of websites accessible only over HTTP, Tor Browser will permit unencrypted traffic between the exit node and destination server, introducing the risk of tampering or eavesdropping. HSTS was designed to protect against these risks. To protect users' anonymity, Tor Browser deletes local data after each session, thus HSTS is effectively disabled for Tor Browser users.

The goal of the CoStricTor protocol is to reduce this risk through the browser forcing encrypted HTTPS connections when the website supports this. Specifically, the protocol will allow Tor Browser to record the HSTS and HTTP status of websites in a series of privacy-preserving Bloom filters which are shared with other Tor Browser users, such that future visitors to the website can force the use of encrypted HTTPS when possible. Not only does this approach mitigate the weaknesses introduced by Tor Browser disabling persistent HSTS records, but it also goes further than HSTS

by protecting Tor Browser users visiting a website for the first time, much like the preload list does.

As we discuss our changes and adaptations from the original RAPPOR protocol it is essential to note the fundamental differences in the layout of our usage. The typical usage of RAPPOR, as described by the authors, is one where data from a *known set* is submitted to the protocol.

In contrast to this, our protocol features an unbounded and unknown set of possible strings, website domains and subdomains. The strings are not known on configuration, but the relevant domain is known to the user when making a query. This setup reduces some complexity in that we do not need to reconstruct the distribution of all strings using regression. This also changes the role of the central server in our setup. A typical RAPPOR server performs regression on this data to produce useful statistics. Our protocol moves the decoding process back to the user so that the user is both a reporter of data and a decoder. This is helpful in maintaining the privacy of users' queries on the data.

6.4.1 Basic Overview

There are three actors in our protocol: the user (acting as a reporter and a decoder), the Tor guard nodes, and the Tor directory authorities.

As decoder, a user must first download the current protocol state, which they can obtain alongside the Tor consensus. Whenever the user visits a website, they query the protocol state. If the protocol indicates the website has been reported as having HSTS, the browser must behave as if it previously encountered an HSTS flag for the website. This means that it may only load the website over HTTPS. Due to the probabilistic nature of the protocol, we allow users to override this restriction by clicking through a warning page. To ensure that the protocol does not permit fingerprinting Tor Browser users, all user must participate in this phase to present identical information across all clients.

Once the website has loaded, if the site does provide HSTS flags, the user's second

role begins. Upon seeing this HSTS flag sent by the server, the protocol encodes the domain name as a submission for the server and perturbs the data according to the privacy parameters to preserve the user's privacy. At the start of the session, the client randomly selects a guard node that is distinct from the user's current guard. All reports for the duration of the session are sent to this node over a new Tor circuit. The guard nodes periodically report their collated reports to a directory authority where it can become part of the latest consensus. The use of a single node for reports allows the use of anonymous tokens for submissions, which mitigates the risk of denial of service, as detailed in 6.4.3.7. Opting out of the reporting phase is possible but lower volumes of reports will degrade the performance of the protocol.

The nature of Bloom filters, combined with the perturbation introduced to achieve local differential privacy, results in a protocol that is probabilistic. Domain names which are reported many times will likely manifest as true positives in queries; domains which are reported less may be lost in the noise. It is not a design goal of this protocol to guarantee full knowledge of websites' HSTS flags to users. Instead, we aim to increase the number of users who benefit from HSTS and increase their security and privacy, while not compromising user privacy.

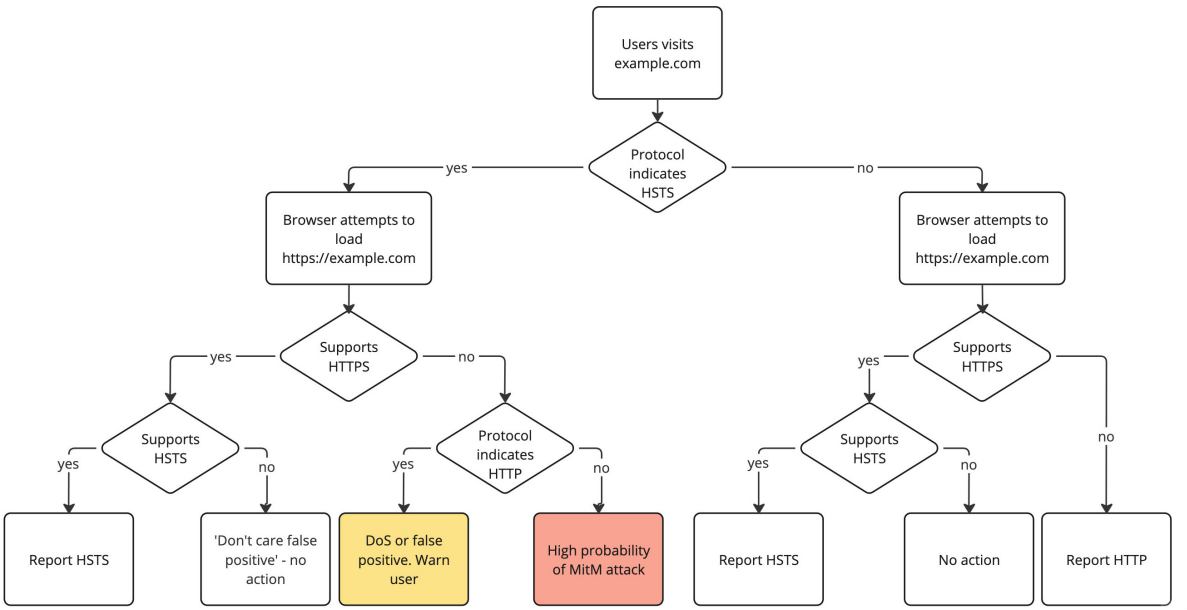


Figure 6.1: Protocol workflow

6.4.2 Threat Model

Similarly to the original threat model against Tor presented in [17], we assume that an adversary can compromise some proportion of nodes. This includes malicious Tor exit nodes that can perform SSL downgrading connections which compromise users' security and privacy. The primary goal of CoStricTor is to provide HSTS to Tor Browser users which can protect against malicious exit nodes without compromising privacy.

Due to the local differential privacy of the protocol, any individual submission does not divulge which domain was visited by the user to the server or to any other user. The immunity to post-processing property of differential privacy means that no adversary can perform further processing on the distributed filters or the individual submissions to reveal the domains that were visited by any individual user.

Although more trusted than other nodes, we assume a small proportion of guard nodes can be subverted. Submissions in the protocol are distributed randomly across all the guard nodes in the network, as clients select them uniformly at random. Directory authorities can therefore expect a similar number of submissions from each guard, any major deviation from this average will be dropped by the directory authority and that data will not be recorded by the protocol.

The Tor network assumes that a majority of directory authorities are honest. Our protocol differs in that a single malicious directory authority can do damage to the protocol proportional to the amount of data reported to them. Just as directory authorities can detect anomalies in the data reported by guard nodes, directory authorities can detect issues in data reported to them by other directory authorities.

In considering the robustness of our protocol, a key failure scenario is denial-of-service (DoS) attacks. False reporting to the protocol could prevent users of the protocol from loading legitimate sites that do not offer HTTPS. DoS attacks through HSTS are a security consideration in the HSTS standard [3, Sec. 14.5] but our protocol introduces another vector.

Although Sybil attacks are problematic for any data aggregation protocol, the anonymity of the Tor Browser makes Sybil attacks on our protocol difficult to detect and prevent. We must, therefore, integrate strong denial-of-service protection, to avoid degrading the protocol's usefulness.

We use two distinct denial of service protections in our protocol. We employ Privacy Pass [239] to prevent adversaries from reporting large amounts of false data in a denial of service attack. This is a rate limiting technique which stops opportunistic attackers from disabling the protocol. We detail this protection in 6.4.3.7. We also integrate DoS protection into the core design of the protocol to mitigate the damage a DoS attack will have if it is successful. This is achieved by focusing on reducing the rate of false positives overall, which is detailed in sections 6.4.3.5 and 6.4.3.6.

6.4.3 Protocol Design

6.4.3.1 Privacy

Tor Browser is designed to protect anonymity and does not save data between sessions which includes HSTS flags. This effectively disables HSTS for Tor Browser users. Our protocol creates a common set of HSTS flags across the entire Tor Browser userbase, so HSTS cannot be used as a tracking vector but users benefit from the security and privacy HSTS provides.

To ensure protection for multiple submissions from a single user, the RAPPOR protocol uses a permanent randomised response which is stored long-term and used for reporting on the same data over time. This is not possible when reporting data with our protocol, as Tor Browser removes user data between sessions. This is, in fact, not an issue precisely because of the anonymity provided by Tor. Different protocol submissions cannot be linked to the same user as, even within the same session they will be made over different Tor circuits.

Thus, it is not possible for an adversary to defeat the differential privacy guarantees over time and the longitudinal privacy protection offered by PRR is not needed. Therefore, our protocol is equivalent to 'One-Time RAPPOR' from [217], despite

multiple submissions of the same data from a single user being possible, and provides the same local differential privacy.

Local differential privacy protects each submission sent to the server. As the aggregate data is differentially private and immune to post-processing, it can be distributed to users. The user performs queries locally, eliminating the need for a private mechanism to query the central data and allowing multiple queries to be performed without a cost to privacy.

6.4.3.2 Expiry of HSTS

HSTS flags come with an expiry time, and as such, it is necessary to accommodate this in our protocol. As the presence of a site in the aggregate counting Bloom filter is probabilistic, and it contains no other information other than the existence of the URL in the set, we have no way to remove websites from the set and no direct way of storing the expiry time stated by a website. The length of expiry time is effectively the longest time the user can go without using the website before the HSTS flag disappears and need to be reapplied. Conversely, the expiry time is also the shortest amount of time a website will have to wait after disabling HSTS before they can switch off HTTPS. In general, disabling HSTS and HTTPS is likely an infrequent occurrence, but it is mandated by the HSTS specification and so must be supported by our protocol.

Expiry times are handled by storing HSTS data for a set period of time, then discarding it. We first decide on a protocol-wide HSTS expiry time which will be used by all entries. We then divide the expiry time into two epochs. Expiry time is implemented by storing two different parallel instances of the aggregate data, each representing one epoch's worth of HSTS entries. Filter set A represents the current epoch and all user submissions are aggregated into it; filter set B represents the previous epoch and it is read-only. When checking if a domain is present, the protocol queries both sets of filters A and B. If the domain occurs in either filter, it is considered present. At the end of an epoch, the read-only filter B is discarded, filter set A is marked as read-only, and a fresh empty set for the new epoch is initialised.

In this system, once a website administrator turns off HSTS, it will disappear from the protocol after at most 2 epochs. The protocol will not produce spurious warnings for websites that have disabled HSTS, assuming the web administrators act according to the specification. Situations may arise where warnings are produced due to HSTS being improperly disabled, but this situation would cause clients using the typical HSTS list implementation to receive warnings as well.

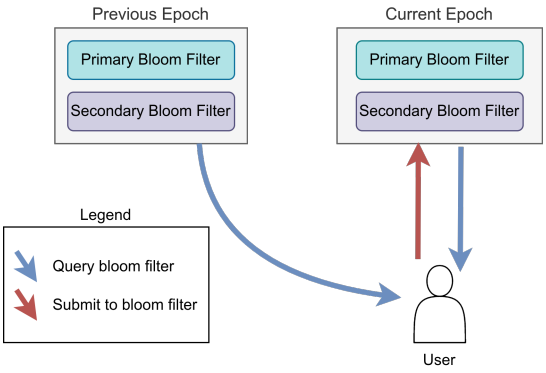


Figure 6.2: Protocol data is stored in two different epochs in order to model HSTS expiry times

Websites with an HSTS expiry time shorter than the epoch length will not be reported, as they may expire or their administrators may disable HSTS before the epoch ends.

We note that this system effectively makes HSTS expiry time discrete. Expiry time in the protocol is either 0 (not recorded) or is equal to the epoch length. The worst case outcome of this is that users may receive less protection if a site is not visited enough between epochs for it to register. Even if the site administrator has a set a longer expiry time, it will not be represented. This however leaves users no worse off, as without the protocol they would not be able to make use of HSTS at all. Similarly, users visiting sites with expiry times lower than the epoch will not receive HSTS protection, but once again they are no worse off as they would not receive protection without the protocol.

The HSTS data we collected shows an epoch value of 30 days provides a good balance between gathering sufficient data and not recording websites with a low

expiry time.

6.4.3.3 Adapting Protocol Parameters to Observed Data over Time

The protocol parameters can be adjusted when a new filter set is initialised at the start of an epoch to better reflect the observed data. Critically, the protocol adapts the Bloom filter sizes based on the number of submissions in the previous epoch, which is used as an estimate of submissions in the new epoch. By using the formula derived from our simulation data we can then calculate the appropriate filter sizes from the estimate.

6.4.3.4 Decision Function

Our protocol needs to output a boolean decision when queried with the domain of a website. In order to produce results, however, a typical RAPPOR instance fits a distribution over the full set of candidate values, which are known ahead of time. This allows for an estimation of the data that was submitted.

In contrast, our protocol works on website domains which are drawn from a theoretically unbounded set of possibilities. Practically, however, only a finite set will be visited by users of the protocol during an epoch, though the set is not known ahead of time. Only the current website being queried client-side is known. Due to these limitations, our protocol must define a more conservative decision function which estimates whether a single domain is in the set of sites reported.

In a setting without noise and no hash conflicts, checking that the count is positive would suffice. Although the mean noise introduced by differential privacy to each count is eliminated from the counts, some counts will be artificially inflated. To account for this, we define the decision function d. It takes a count of bits n, the number of insertions to the filter i, and the estimated number of distinct websites reported w and outputs a Boolean decision. Additionally, a decision multiplier c is introduced to allow for tuning the function's output to be more or less sensitive to noise.

$$d(n,i,w) = n > \frac{i}{w} \cdot c$$

The decision multiplier is set empirically based on the simulation results described in Section 6.5.

6.4.3.5 False Positives

Bloom filters are at the core of our protocol. They are probabilistic data structures which introduce a rate of false positives. This is further exacerbated by the noise introduced for differential privacy. In the case of this protocol, a false positive indication would mean a user who erroneously believes a website has enabled HSTS when, in fact, the website does not set an HSTS flag.

A false positive generates two possible situations. The website may have HTTPS available: this is the less problematic situation; while the website may never have mandated HTTPS for all its visitors, the user can still access the website over HTTPS and they will not have any interruption to their browsing experience. In fact, as Tor Browser has recently implemented HTTPS-Only Mode by default, there will be no difference to users' browsing experience for sites which implement HTTPS. We term this situation a 'don't care false positive'.

Scenario two is that the website does not, in fact, support HTTPS. This is an undesired outcome for the protocol. According to the HSTS specification, the website should now be blocked and fully inaccessible to the user. We term this outcome a significant false positive. An attempted DoS attack would aim to produce this outcome for target websites. Given that our protocol has a false positive rate, we instead insert a warning page which can be overridden by the user. This allows us to inform the user that they are likely at risk, but also gives them the option to continue if this is a clear false positive.

Careful parameter selection can reduce false positives while maintaining strong utility of the protocol in upgrading websites to HTTPS. Section 6.5 describes the results

of simulating the protocol and, in particular, the effects varying different parameters have on the results.

6.4.3.6 Double Bloom Filter Construction

To further reduce false positives, CoStricTor not only records sites which report HSTS usage, but also separately records sites which do not implement HTTPS. Thus, the protocol separately collects data for domains with HSTS and domains without HTTPS in two aggregate counting Bloom filters. We term the aggregate filter that collects HSTS data as the 'primary filter'; the aggregate filter for non-HTTPS domains is termed the 'secondary filter'.

Recording complementary data is possible as websites are divided into three disjoint sets:

1. Websites that support HTTPS and HSTS
2. Websites that support HTTPS but do not have HSTS enabled
3. Websites that do not support HTTPS

Note that websites that support HTTPS but do not have HSTS enabled, which are the majority of websites, do not need to be tracked as they can only produce don't care false positives. Therefore, by capturing two reduced subsets of all possible websites, we can efficiently create complementary data that can be used to detect false positives and is resilient to fake submissions.

The addition of the secondary counting Bloom filter changes the submission process by making users submit websites without HTTPS, alongside those with HSTS. The secondary filter is used for confirmation when a result of the primary filter blocks the user from accessing a non-HTTPS site, that is, when a query to the protocol indicates a website as having HSTS by querying the primary filter but the website does not support HTTPS when the user attempts to load it. In this scenario, the secondary filter is queried. If the site is found in the secondary filter, that is, if it has been reported as having HTTP previously, CoStricTor concludes that a false

positive must have indeed occurred. If the website is not found in the second filter, the protocol concludes that HSTS was present previously and that a man-in-the-middle downgrade attack is occurring. This allows for an appropriate warning to be displayed to the user.

The double filter construction also increases resistance to fake submissions. As described in 6.6.1.1 detection of sites in the secondary filter attackers ability to block users from sites. The second filter introduces a more refined approach to potential false positives, practically reducing the number of significant false positives, and produces a better failure state.

6.4.3.7 Denial of Service

All submissions to the protocol must be done through the Privacy Pass [240] mechanism, currently an IETF draft but supported by large platforms such as HCaptcha [241]. This mechanism issues a set number of anonymous tokens after a challenge is completed. Challenges can be interactive, such as CAPTCHA, or non-interactive, such as attestation or authentication, neither of which are suitable for Tor Browser due to privacy concerns. An alternative to CAPTCHAs, which disrupt the user experience and introduce a barrier to legitimate submissions, is computational puzzles. Client puzzle introduce a minor computational cost to submissions to mitigate DoS [242]. They have a long history and were first proposedin response to connection depletion attacks, such as SYN flood attacks, and then applied to a range of DoS attacks. Variants using timed cryptographic primitives, such as verifiable delay functions, have been proposed [243]. These have the advantage of being able to run in the background and imposing a low cost to legitimate browsing patterns while an attempt at a DoS will require solving a large number of puzzles.

The Tor guard node selected by the user for the session will issue anonymous tokens, which are redeemed to submit data. This effectively rate limits users who submit to the protocol, slowing down adversaries who wish to submit bad data. It also has the additional affect of forcing users to submit to the same guard node, which limits adversaries ability to spread their attack across multiple nodes, mak-

ing their attacks more detectable. The number of tokens issued for each challenge (CAPTCHA or client puzzle) can be adjusted to trade-off increased protection with worse user experience.

6.4.3.8 Warning Users of HSTS Failures

A critical consideration for this protocol is the presentation of information to users. When CoStricTor determines that a website has HSTS based on the primary filter but a secure connection cannot be established, two possible situations arise that require informing the user. The secondary filter, which encodes the lack of HTTPS, is used to discriminate between the two situations that elicit a warning.

If a website is present in the primary filter but missing in the secondary filter, the protocol concludes that there is a risk that HTTPS was stripped. The protocol presents a full-page warning to the user similar to the browser's built-in HSTS warning. Because of the probabilistic nature of the protocol, we propose using a modified version that allows the user to continue to the site, unlike the typical HSTS warning which blocks access. It will briefly describe to the user that this warning is based on crowdsourced data and that there is a low, but non-zero possibility of errors.

In the scenario that the website is present in both filters the protocol does not take any action and the standard HTTPS-only mode warning page is displayed.

6.4.3.9 Reporting Bad Exit Nodes

An additional utility provided by the protocol is identifying potential bad Tor exit nodes. Whenever a site is detected as having HSTS and it does not support HTTPS, this indicates a potential SSL stripping attack. This attack may have been perpetrated by a bad exit node and thus each detection counts as some evidence an exit node is bad. By reporting the identity of every exit node involved, Tor's operators can identify bad exit nodes by the volume of reports received. Due to the random assignment of exit nodes to users, there will be a baseline level of reporting due to SSL attacks external to Tor and false positives. If exits nodes will have a proportionally higher level of their traffic reporting SSL stripping, it serves as strong evidence that the node is conducting an attack. Reports of exit node identities leak none of

the users' information and thus can be reported via a Tor circuit to a centralised server operated by the Tor Project.

6.4.4 Protocol Specification

We now describe the behavior of our protocol, including the scenarios in which the protocol activates, as well as the exact details of how the protocol operates. Figure 6.1 is a flowchart of the decisions the protocol makes as a user visits a website, showing when decoding and submission occur as well as when warning pages are displayed.

6.4.4.1 The Query Algorithm in Detail

The initial phase of the protocol occurs at the start of a Tor Browser session. Just as the Tor Browser receives the latest consensus data from the directory authorities, the browser shall also receive the latest protocol data for detecting HSTS.

This data contains a set of 4 counting Bloom filters of length k containing noisy counts along with the count of submissions for each counting Bloom filter. The current count is necessary for the client to adjust the data to compensate for the added noise. Upon receiving this data, the browser performs the following correction for each of the 4 filters, where N is the current insertion count for that filter, c is the initial filter data, and t is the adjusted data.

$$t_i = \frac{c_i - pN}{q - p}$$

Performing this correction step compensates for the differential privacy noise added, and gives us more accurate counts. This concludes the steps required upon starting a Tor Browser session.

The main query process occurs when the user visits a website. At first, Tor Browser will automatically check if the domain is present in the HSTS preload list. If it is present, the protocol will not run as neither checking nor submitting that website will be of benefit. Otherwise, the browser will continue by running CoStricTor.

Before the request is sent to retrieve the web page, the browser checks for the presence of the domain in the primary counting Bloom filters of both epochs. To do this, the relevant counts are obtained from the filters extracting the counts that match the bits where a domain would be inserted. These approximate the number of submissions for the domain.

The protocol's decision function uses the counts to determine if they indicate the presence of the item in the aggregate counting filter.

As discussed earlier, this function is as follows:

$$d(n,i,w) = n > \frac{i}{w} \cdot c$$

where n is the count, w is an estimate of distinct websites submitted, i is the total number of insertions and c is a constant correction factor which is applied. The estimated number of distinct websites submitted is calculated by matching the number of submissions against a standard zipfian curve. As we assume all website visits follow this zipfian distribution, the estimate will be the highest rank on the curve that will appear at least once given the number of insertions.

If the website does not appear in either of the primary Bloom filters the connection continues as normal. If however, the website is present, we assume the site has HSTS. Subsequently, the browser will attempt to load the webpage via a HTTPS connection. If the website loads correctly, we need not take any further action. If the site does not load via HTTPS, the protocol must now employ the secondary counting Bloom filters. If the domain is present in either of the secondary filters, this indicates some form of false positive result. We cannot rely on the output of the protocol in this instance so we direct the user to the standard HTTPS-Only mode warning page. If the domain is not present in either of the secondary filters, we have confirmed that this website usually implements HSTS and thus the user may be at risk of a man-in-the-middle attack. We display a custom warning page to the user informing them of the risk of proceeding to the webpage.

6.4.4.2 The Reporting Algorithm in Detail

There are two scenarios which need to be reported in CoStricTor: the user loads a domain that has HSTS enabled or a website that does not support HTTPS. When either of the scenarios occur, the protocol checks if the domain exists in the HSTS preload list and ignores it if present. The protocol will also not report any HSTS websites that have an HSTS expiry time shorter than the length of an epoch.

A report will then only be made if the domain has not already been reported this session. This is prevent over-reporting when a website is visited many times.

To construct a domain report for either scenario, the protocol initializes an empty Bloom filter. The appropriate domain is inserted into the Bloom filter. Local differential privacy is ensured by perturbing each bit of the Bloom filter according to the privacy values p and q. This process is identical to constructing the IRR in RAPPOR.

This report is then sent through a new Tor circuit to the users *secondary guard node*. This guard node is randomly selected at the beginning of the Tor Browser session and must be distinct from the users current guard in order to prevent submissions from being linked to the user. The client constructs a new Tor circuit for every report sent to this guard node in order to stop different submissions being linked to the same user. Every report received by a guard is added to their counting Bloom filter data structure containing occurrences of each of the bits in the bit array for the current epoch. The Bloom filters corresponding to the previous epoch are read-only. As part of Tor's normal operations, guard nodes will query a directory authority server to obtain the latest consensus periodically. While obtaining the latest version of the consensus the guard node can now report their Bloom filter data structures to this directory authority, and the directory authority will provide to the guard the latest complete version of the Bloom filter data it has obtained from the other directory authorities. Directory authorities vote every hour on the latest version of the consensus and at this point they will also collectively sum the Bloom filter data they have received to obtain the definitive Bloom filter data. This version of the

Bloom filter data is then distributed as part of the consensus to Tor clients, via their guard node.

6.5 Evaluation

To find appropriate parameters for CoStricTor and to evaluate its effectiveness, we simulated the protocol. The simulation was based off the sample implementation of RAPPOR [244] but rewritten in Golang to favor simulation speed and adapted to our protocol design. It has been open-sourced [245].

It allows the testing of all system parameters and provides us with specific results on two key metrics: number of sites correctly upgraded by the protocol, and number of sites with incorrect warnings displayed by the protocol. The simulation parameters are: Bloom filter size, the number of reports, the number of websites from which reports can be drawn, the number of additional sites checked, and the privacy parameters p and q, from which the overall ε of the system is computed. The simulation uses the Majestic Million list of the one million most popular domain names.

The simulation uses a Zipf distribution to estimate the popularity of different websites which can be submitted. Informally, we know that certain websites are extremely popular and see many more visitors, whereas a huge number of sites are far less popular and are very seldom visited. In [192], the actual distribution is estimated to be a power log distribution. This is also backed up by [246] and [247] which confirms the ubiquity of power log distributions in Internet measurements, but, more specifically, the applicability of a Zipf distribution, which is a special case of power log, to website popularity. Past measurements by Nielsen [248] have also confirmed this pattern of website use. Whenever a user submission is being simulated, we draw websites from our list of domain names through a Zipf distribution so that websites appear in submission in a manner that mirrors the real world. Finally, to better simulate potential false positives which occur for websites that are never submitted, the simulation also checks an additional list of websites not submitted by any user against the counting Bloom filters produced by the protocol. For

example, if we are considering 1000 websites in the protocol, we can run a simulation over this data, and then check an additional 10,000 websites to check for false positives.

Our protocol simulation works as follows: the n most popular websites are taken from the Majestic Million list [65]. We use estimated proportions of HSTS and HTTPS sites to randomly designate websites as HSTS, HTTPS, or HTTP. We then randomly sample websites from the list with replacement. Samples simulate user submissions to the protocol. If the sampled site has been designated as having HSTS, we add this to our local Bloom filter exactly as described in the protocol, and likewise if the website was designated as having HTTP only.

After these steps, our Bloom filter data can be queried to simulate users' results when using the protocol.

6.5.1 Simulation Parameters

6.5.1.1 p, q the Differential Privacy Parameters

These parameters directly affect the amount of noise added to user responses. They must be set at a level that gives the appropriate differential privacy guarantees, while retaining utility in the submissions. We evaluate a range of different ε values to ascertain accuracy of the protocol with different privacy guarantees.

6.5.1.2 Number of Sites Modelled

The number of sites is a key number in our parameter selection. The more sites represented by the protocol, the larger Bloom filter is needed. The key difficulty is that the system can never know exactly how many sites are currently being represented. By definition of this being a privacy-preserving system it is not possible to know this number, but we can estimate it using the total number of submissions to the protocol in a period of time.

6.5.1.3 Estimated Number of Submissions

The number of submissions the protocol receives will vary over time. To simulate the protocol, we must provide an accurate estimation of the number of submissions

that will be received in one epoch. Our simulation runs exactly one epoch of the protocol, and so we must first decide the epoch length in order to estimate the number of submissions. Epoch times that are too long may result in website administrators being unable to effectively remove their HSTS status. More specifically, if HSTS is turned off on a website, there may be a significant period of time where the site is still registered with the protocol as deploying HSTS. Epoch times that are too short provide less utility. Destroying the protocol's data very frequently means that the density of submissions for any one website will be lower, and a lower number of websites will be recorded by the protocol as having HSTS. In the simulation, we use 30 days as our epoch length. From our earlier data gathering, we determine that 83% of sites reporting HSTS use an expiry time of over 30 days. Additionally, 30 days of submissions from users provides plenty of time to allow representation of a large number of sites. From this, we can now estimate the number of submissions the protocol receives in one month. Given that the protocol is intended to be integrated into Tor Browser, we can assume all its users would use the protocol to report websites. In a period of 30 days, Tor receives approximately 90 million connections from users. We assume that each of these connections visits 1 website. This is a weak assumption as many users will visit more than one website. Therefore, over a 30-day period, we can expect approximately 90 million queries to the protocol. This is the figure used to run simulations on the protocol.

6.5.1.4 Estimated Proportion of HSTS Sites

Once again, the actual number of HSTS sites in the population of websites recorded in our protocol is a key figure. The proportion of HSTS sites in our population determines how much of our "estimated number of sites" affects the size of our Bloom filter. For example, we may have 100 000 sites in our population, but if only 1% deploy HSTS our Bloom filter only needs space for 1000 entries. For our simulations, we use the figure 15% which we derive from the proportion of websites with HSTS our data collection revealed.

6.5.1.5 Estimated Proportion of Non-HTTP Sites

As above, this proportion determines how many of the sites in the population are recorded by our secondary Bloom filter. Consequently, a higher proportion of non-HTTP sites will require a larger secondary Bloom filter. Considering the distribution of site popularity, our protocol is much more likely to deal with popular websites and less likely to be faced with less popular sites. Given that a strong majority of popular websites use HTTPS, we can expect a relatively low percentage of sites considered to have no HTTPS available. For our simulation, we estimate the proportion of non-HTTP sites to be 20% [232] . A mismatch between this estimate and real proportion would lead to a poorly sized secondary Bloom filter, affecting accuracy of false positive detection. However, a real deployment of the protocol can dynamically update the secondary filter size based on the number of submissions received in the previous epoch.

6.5.1.6 Bloom Filter Size

Larger Bloom filter sizes, result in fewer false positives overall. If the filter size is larger for the same number of entries, there will by definition be less overlap in the bits of the Bloom filter and fewer false positives accordingly.

6.5.1.7 Decision Function Correction

As discussed, the protocol requires a decision function which includes a multiplicative correction term. We estimate the correction term by running the simulation with various candidates for the correction term until we arrive at the value which produces the best results. These values are 0.045 for querying the primary filter and 0.07 for querying the secondary filter. The values scale with with the proportion of websites considered, meaning that, for the simulation, the secondary filter has a higher value but a lower actual threshold. This biases the simulation towards avoiding false positives. The result is that false positives in the primary filter can be mitigated by the secondary filter more easily.

6.5.2 Results of the Simulation

We run the simulation with a range of different numbers of sites being considered. In a simple case, we can consider 500 of the top websites. Using modestly sized Bloom filters we can easily provide accurate results with a high number of upgrades, and a very low number of false positives. The primary goal of our simulation was to maximise the number of sites considered without sacrificing accuracy. As we discussed, there is effectively an infinite number of sites that could be visited, but, in practice, we can expect that only a small number of sites are visited with high regularity and by many users. The higher the number of sites considered, the closer we can approximate the actual set of sites that the protocol might be asked to record in a real deployment. The primary restricting factor on this is the number of submissions made to the protocol in a certain time period.

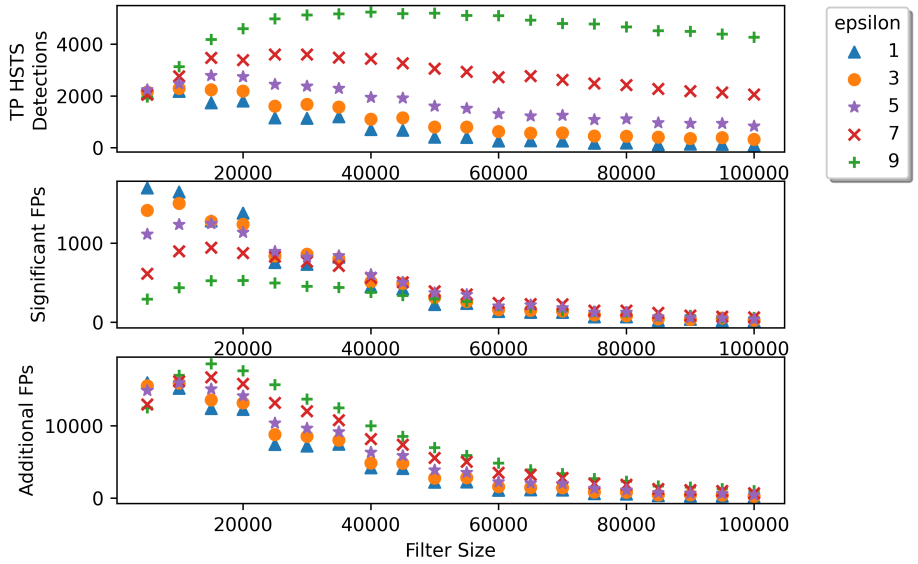


Figure 6.3: Increasing filter size gives more accurate results until an optimal point past which accuracy declines

The first task of the simulation is to establish which size of Bloom filter is appropriate for the protocol. We can see in Figure 6.3 the results of the simulation when considering 50 000 websites while varying the size of the filter as well as values of ε. At very low Bloom filter sizes performance is poor, but quickly increases to a maximum. As filter size increases beyond this, the number of upgrades slowly declines despite improving false positive rates. This can be explained by observing

that, for a fixed p, the number of 0 perturbed into 1 increases roughly with the size of the filter. Increasing the size of the Bloom filter increases the overall chance that, any zero bit in a report is perturbed. This implies an optimal filter size exists for any particular setup, given an accurate estimation of the number of sites being covered and the number of submissions that will happen.

The filter size affects the amount of data which must be distributed to every Tor client as well as the size of the data which is reported to guard nodes. From Figure 6.3 we estimate 40 000 as a typical filter size for the system. We conducted additional simulations to extract the exact Bloom filter data of several runs of the protocol. This allows us to check the actual size of the data which needs to be transmitted. Our benchmarks show the total size of the 4 filters when transmitted is on average 130 kB for a filter size of 20 000 and 260 kB for filter size of 40 000, when compressed using the standard Brotli algorithm; the mean size of compressed submissions is 1.4 kB and 2.8 kB, respectively.

The next consideration is the privacy parameters. Differential privacy is calculated in the same way as RAPPOR: $\varepsilon = \log(\frac{q(1-p)}{p(1-q)})$ from which we derive

$$p = \frac{q}{(1-q)e^{\varepsilon} + q}$$

Recall that p is the probability of any true 0 bit being reported as a false 1, and q is the probability of any true 1 bit being reported as a 1. We explore the space of values of p and q for a set value of ε. The results of these simulations (Figure 6.4) show little variation, which indicates that the choice of p and q need only achieve a desired value of ε and that p and q are not a target for optimization for a fixed ε.

In order to produce a clear idea of the number of sites that we can accurately represent, we compare the maximum number of sites we can consistently represent for different values of ε. For this, we must also obtain the best-suited value for filter size for each combination of ε and number of sites being represented. We define a

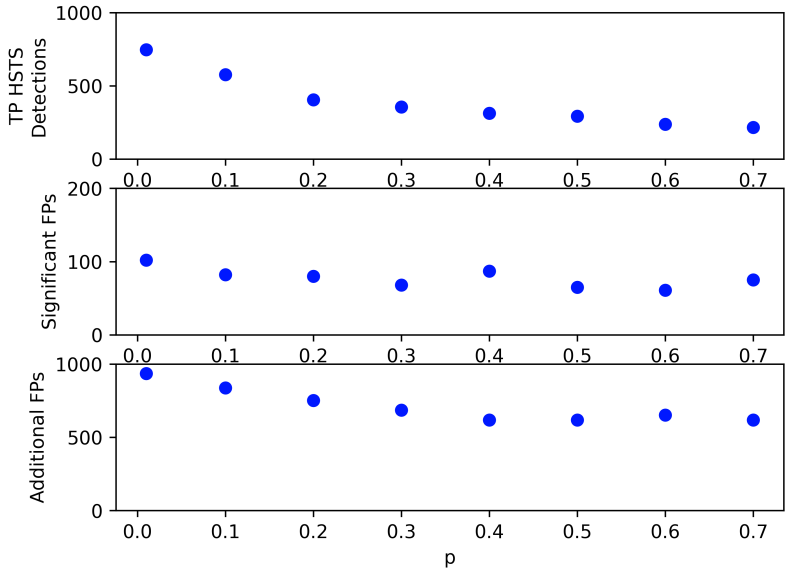


Figure 6.4: For the same ε, differing combinations of p and q produce negligible difference

function f which produces the optimal value of Bloom filter size s for a given ε and w, the number of sites being considered, from a set of candidates sizes S, using the outputs of multiple runs of the simulation:

$$f(w, S, \varepsilon) \rightarrow s$$

In this context, we can consider the simulation a function that given w and a pair of values, $s \in S$ and ε, produces values: u, the number of HSTS upgrades, and d, the number of HTTP sites that are falsely blocked.

If the protocol returned perfectly accurate results we could set f to return the maximum number of upgrades as long as false positives are zero, but this is, of course, impractical. By allowing a small number of false positives, we can obtain much much better results for number of upgrades. The optimal definition of f depends on the associated costs of a spurious warning page and the benefits of an HSTS upgrade. We consider finding an optimal definition for Tor Browser to be future work. Choosing a conservative definition of f can keep false positives very low, but greatly restricts the number of upgrades we can provide. Similarly, we could also

define f to allow much more false positives which would allow a much greater number of upgrades but would face users with a much greater number of false warning pages.

To evaluate the protocol, we choose the following definition of f, which ensures that the number of false positives is at most a quarter of the number of upgrades.

$$f(w,S,\varepsilon) = \underset{s\in S}{\text{argmax}_u}\ \text{simulation}(w,s,\varepsilon), \text{ subject to } d < \frac{u}{4}$$

6.5.2.1 Final Results

By applying f to a range of inputs we obtain data showing the best results our protocol can produce for any value of ε and the number of sites being considered. In Figure 6.5 we can see the results from epsilon values 1–9 and sites values up to 160000. This data is the culmination of our protocol evaluation and shows the best results we can achieve given our assumptions.

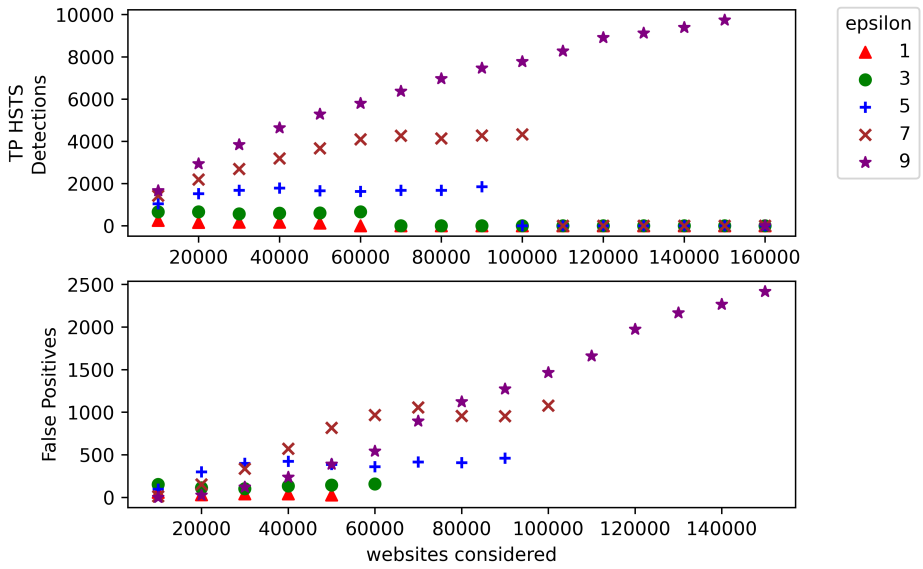


Figure 6.5: Depending on the value of ε, the protocol can only a model a certain numbers of websites before false positives are too high. In this figure an excessive ratio of false positives to upgrades reduces the upgrade number to 0.

It is important to note that even in an ideal situation, the number of sites upgraded does not scale linearly with the number of sites considered. This is a result of the

Zipfian distribution of popular sites, which exhibits a long tail. If we consider 1000 sites, many of them will be highly popular sites that receive many visits and will be represented well in the protocol. As we add sites, however, another 10000 sites will have a dramatically lower percentage of visits than the top 1000.

6.6 Discussion

The simulations show that the protocol can model up to 150000 websites with reasonable differential privacy guarantees, and potentially much more if ε is relaxed further. The protocol provides 10000 upgrades to HSTS with 2500 false positives. The ratio of upgrades to false positives can of course be reduced with a more conservative definition of the selection function f. The limiting factor is the number of submissions; with a greater number of users of the system, we can accurately model more websites which means we can more closely approximate the actual number of websites that are visited by users. The protocol will also perform better as the number of HSTS enabled sites continues to increase.

The protocol can bring quantifiable security benefits to Tor Browser users. When CoStricTor accurately presents a site as having HSTS, it provides all of the benefits that HSTS usually brings, including removing the ability for an adversary to perform SSL stripping attacks. Although Tor Browser now implements HTTPS-Only Mode by default, if an adversary attempts an SSL stripping attack, the user will be presented with a more accurate warning page indicating to them that the site once had HTTPS and no longer does, replacing the HTTPS-only mode warning which simply states that the site currently does not have HTTPS. Users will never be faced with more warning pages than they would see without using the protocol.

HTTPS-Only warnings have relatively frequent occurrence as a user browses the web and they can only provide a non specific warning to users, sites which do not implement HTTPS are not necessarily malicious and often the best course of action for the user is to click through the warning. In contrast, a HSTS warning from our protocol shows the user that there is a real danger from the site they are about to visit.

The relatively low false positive rate ensures that users can have a high confidence there is a danger from this site and it is unlikely to be something they should click through.

We note that not only does our protocol overcome Tor Browser's inability to use HSTS, it also protects Tor Browser users visiting a website for the first time. HSTS, by itself, employs a 'trust on first use' paradigm where users are not protected the first time they visit a website. CoStricTor overcomes this issue since other users may have already reported the website as implementing HSTS.

The benefits of the protocol come at a cost to increased traffic on the Tor network to aggregate and distribute the filters, and additional complexity in operating a guard node. The communication cost depends on the filter size. Our benchmarks show a user needs to initially download 130 kB of data for filters of size 20 000 or 260 kB of data for filters of size 40 000, as these sizes provide adequate performance. For comparison, the compressed size of the consensus is approximately 580 kB. Although this increases the size of the initial download, the filter set for the previous epoch is read-only and non-specific, therefore can be cached across sessions, halving the size of subsequent downloads to update the filter set for the current epoch. The overall increase in Tor network traffic from submissions and filter downloads is not significant compared to the total traffic over the Tor network which is over 250 Gbit/s [249], although it increases the load on directories and directory mirrors. Additional traffic between guard nodes and directory authorities is also relatively minor.

6.6.1 Limitations

6.6.1.1 DoS Attacks Can Render the Protocol Ineffective

The CoStricTor protocol is resilient to some forms of denial-of-service attacks. Opportunistic attackers will be deterred by the Privacy Pass mechanism. Attackers with more resources can still overcome this rate limit and attempt to perform a DoS attack.

There are two potential attacks the CoStricTor system. Attackers can either submit false entries to the primary Bloom filter, or to the secondary Bloom filter. Erroneous entries in the primary filter can make users avoid visiting certain websites by producing false warning pages on non-HTTPS websites. This attack can only be successful if the website being targeted has not been sufficiently submitted to the secondary filter, otherwise any false results generated from the primary filter will be disregarded, which makes it ineffective against popular websites. Erroneous entries in the secondary Bloom filter can negate the utility of the protocol and potentially allow SSL stripping attacks to occur. This second kind of attack cannot be avoided, but is made more difficult by the implementation of the Privacy Pass mechanism as described earlier. If an attacker sends many false submissions to the secondary filter about a website, the protocol will believe that this website does not implement HTTPS and therefore any hits in the primary filter are false positives. The Privacy Pass mechanism means that an adversary must complete a challenge to make submissions. Although the number of submissions that can be made per challenge solved can be varied, we estimate that 20 submissions would be sufficient for a typical browsing session. Using our earlier assumptions about Tor user activity, we estimate that in order to cause a website to appear in the secondary Bloom filter, 315 000 submissions are needed to be made. Given 20 submissions and assuming CAPTCHA challenges, we calculate that an adversary must complete 15 750 CAPTCHAs in order to perform this attack. At a rate of $2.99 per 1000 CAPTCHAs [250], it would cost an adversary $47.10 to perform the attack. A secondary filter that was entirely full of false reports would reduce the protocol's utility to zero. In this scenario, no utility is gained from the protocol, but crucially users are no worse off than if the protocol was not present. An attack on the primary Bloom filter would require 202 500 submissions and would cost approximately $30.27. Any attack performed would only be effective for a single epoch and would need to be repeated after it expires. This additional cost must be borne by the attacker to perform a SSL stripping attack whereas previously without the use of the protocol there was no additional cost or barrier for performing attacks.

Malicious guard nodes could also be used to perform these attacks by avoiding the Privacy Pass restrictions. Currently, there are approximately 3000 guard nodes, this means that any one malicious guard node can only alter approximately $\frac{1}{3000}$ of the protocols data, giving them limited power over the protocol. Similarly, a malicious directory authority has the ability to have approximately $\frac{1}{9}$ effect on the protocol. Large deviations from the expected data, such as a large spike for a single value, can be detected and discarded, helping mitigate harm to the protocol by limiting the impact of malicious nodes and DAs.

6.6.1.2 Only Effective for Very Popular Sites

As we aim for a strong level of differential privacy for users of the system, the protocol by definition cannot show a site as having HSTS unless there are a large number of reports made. Additionally, as we have discussed, the Zipfian distribution of website visits means that the most popular websites receive the vast majority of traffic, and the number of visits falls dramatically as we consider more websites. The result of both of these factors means that only a certain number of popular websites can be modelled by the protocol. Regardless of any other parameters we use for the protocol, the number of sites we can positively report as having HSTS is limited by the level of differential privacy we require and the popularity of websites we want to represent. This also means that there is no risk that users will be identified by their visiting a very unpopular/niche site, as there will not be enough reports of this site to cause it to be represented by the protocol.

6.6.1.3 Lack of Country Specific Data

As highlighted earlier in this work, HSTS and HTTPS deployment can vary substantially based on country. The data collection work described here showed that there is a discrepancy between HSTS across countries, however, it was not possible to find country specific data beyond the 100 most popular websites in these selected countries. To ensure that our protocol remains accurate for the sets of websites being accessed by Tor Browser users in diverse countries it would be necessary to obtain more comprehensive data on the deployment level of HSTS in these different regions. Until this data is available, we cannot be fully confident that the results

shown are generalisable to this wider set of Tor Browser users.

6.6.2 Future Work

6.6.2.1 Real-world Evaluation

To establish the viability of the protocol, it was evaluated through a set of simulated scenarios. A full evaluation of the protocol would require real-world usage data. Although the ultimate aim, deploying the protocol in Tor Browser without real-world testing is not practical. As an intermediate step, we aim to create a web browser add-on that runs the protocol. This will enable us to test different aspects of the system. For example, does the protocol introduce any noticeable increase in latency as the system computes the HSTS status of websites?

6.6.2.2 Warning Pages

This system introduces additional warning pages which can be shown to users as they browse the web. There is significant research on creating effective warnings for common web browsing scenarios [183, 191, 196, 208]. SSL/TLS errors cause the browser to display well-studied, thoughtful, and effective warnings which accurately inform the user and enable them to make the best decision. The new warnings introduced by the protocol must also conform to this high standard and we wish to perform a user study to show users are correctly informed to make the correct decisions.

6.6.2.3 Appropriate Proportions of False Positives

The CoStricTor protocol provides HSTS data to users which is correct with a certain probability. False positives can always occur, however, there are also trade-offs which must be decided when implementing the protocol. We outlined the function f which decides the best Bloom filter size for a given set of other parameters. f can be defined to produce very high positive results if it is given a lower tolerance for false positives, however, if the tolerance for false positives is reduced, the protocol still produces positive numbers of upgrades at this lower level. We aim to configure f for a real Tor Browser deployment by conducting a qualitative survey of Tor experts. This work will allow us to set a value that provides the most benefit for

users, whilst also not burdening users with tedious warning pages. This is crucial not just for user experience, but also to avoid introducing warning fatigue which can devalue the usefulness of our warning pages if they are displayed too often.

6.7 Related Work

6.7.1 Adaptations of RAPPOR to Unknown Dictionaries

Some attempts have been made in the literature to adapt RAPPOR for different purposes. In particular, in [251] the authors attempt to use RAPPOR with an unknown set of strings through the reporting of n-grams of submitted strings. This technique, while effective, is unnecessary in our scenario, since our candidate strings become known when the user needs to query the protocol.

6.7.2 Private Data Collection in Web Browsers

Prio [252] is a protocol for gathering aggregate data in a privacy-preserving manner. It is in use in Mozilla Firefox. Prio uses multiple partially trusted servers for storing shares of the reported data, which can then be aggregated to reveal the aggregate data. The privacy of users' data is preserved as long as one of the n servers involved is not compromised. Prio uses secret shared non-interactive proofs to ensure that submissions meet set correctness parameters. Prio is limited to only encoding numerical data which can be aggregated between servers.

A similar data aggregation system was deployed in Microsoft products and is detailed in work by Ding *et al.* [253]. The algorithm particularly attempts to address privacy for metrics that are reported repeatedly. In RAPPOR the issue of reporting data repeatedly is addressed by the memoisation step. This new system attempts to avoid this problem in a more efficient manner. Ultimately, it is not relevant to our protocol as the protocol can never report data repeatedly.

Prochlo [254] is a mechanism introduced for privacy-preserving data collection in the Google Chrome browser to improve on the utility of RAPPOR on many types of queries. Prochlo is designed as a three-step pipeline: encode, shuffle, and analyse, with each step intended to be performed by a separate entity. Encoding on end-

users' machines controls the encryption, granularity and if local differential privacy is desired, randomness of the data. Shuffling strips data of metadata and acts as a mix that batches reports over a long time period. Randomised thresholds for batches or randomly removing data at this step can introduce differential privacy. Shuffling must be performed on trusted hardware or on otherwise trusted machines. Finally, the analyse step is performed on the batched data to obtain the aggregate data being sought. Centralised differential privacy can be introduced to the analysis step.

STAR [255] is a system for threshold data aggregation intended for use in the Brave browser. Each item becomes available to the collector only if it is submitted at least k times. The threshold is based on k-of-n secret sharing of an encryption key derived from the data using a verifiable oblivious pseudo-random function (VOPRF) run by a third party. Each participant with the same data can obtain the shared encryption key and its shares from the VOPRF.

The design offers protection for inputs drawn from high entropy distributions where a malicious server cannot pre-compute the output of the VOPRF but, as the encryption key is derived from the input, a malicious collector can subvert the threshold for any known values. The design goals of STAR explicitly include correctness, with exact counts of submissions over the threshold, rather than approximate values so it does not provide local differential privacy. Similarly to CoStricTor, STAR can use the TOR network to prevent linking submissions to individuals.

PrivEx [256] is a private data aggregation system that is specifically designed for use in anonymous communication networks, in particular Tor. PrivEx can use either secret sharing or distributed decryption mechanisms as well as differential privacy to ensure data can be collected anonymous and in a way that is resistant to adversaries. In particular PrivEx is designed to maintain counts of particular websites being visited in the anonymity system.PrivEx maintains a list of counters for these websites that are under observation. Typically these are designed to be websites that are known to be censored and as such that anonymity network administrators may desire analytic data on how their network is being used. Once the protocol has been

initiated, the list of websites is static. The protocol cannot model an infinite set of websites like RAPPOR can which ruled PrivEx out as a candidate for building this protocol.

6.7.3 Other Additions to Tor Browser

Dahlberg *et al*. constructed a mechanism to add new Certificate Transparency capabilities to Tor Browser [257]. The new protocol for probabilistic reporting of Certificate Transparency failures mirrors our own new protocol for Tor Browser. Much like our HSTS protocol, this certificate transparency protocol requires reporting data from Tor Browser to centralised locations, and additionally required simulations to be run in a similar style to our own protocol.

6.8 Summary

CoStricTor, the protocol we have presented provides real value for Tor Browser users by allowing users to share HSTS data for a large number of popular websites, introducing these protections where they were previously unavailable. This protects against malicious Tor exit nodes attempting to downgrade connections and tamper with data, where previously Tor Browsers' lack of HSTS support left users vulnerable to attacks. We show that false positive rates are low and users will be infrequently interrupted by erroneous warning pages. The protocol also maintains user privacy which is a key concern of Tor users. Our simulations show that accurate results can be given for over 150 000 websites with 10 000 HSTS upgrades, depending on the level of differential privacy required. We also provide a valuable detection system for bad Tor exit nodes, preventing future attacks and assisting with the health of the Tor network as whole.

Chapter 7

Conclusion

7.1 General Conclusions

Anonymity is a very useful property on the internet, but one can only ever be truly anonymous amongst a group of other individuals. Anonymity tools like Tor or mixnets require large, diverse sets of users to properly anonymise their users. As a result, usability has become an incredibly important factor for anonymity software. Allowing anonymity software to be accessible to wider groups has been a key goal of Tor Browser. For low-latency anonymity systems like Tor, usability is already relatively advanced; Tor Browser is a mature and well-studied piece of software that focuses on usability. High-latency mixnets, on the other hand, have not developed widely used applications and mainly still struggle with latency as a usability problem.

The thesis has discussed in its four main chapters, different methods for further increasing the usability of both high and low-latency anonymity systems. The majority of the advancements made here apply to both types of anonymity systems and can increase the usability of a mature application like Tor Browser, or could be applied to new applications for high-latency anonymity networks.

The work done in this thesis further cements the importance of usability work for anonymity systems. More specifically, the projects in this thesis addressed this

using indirect methods. Typical methods for improving usability in anonymity systems, particularly Tor, addressed the most obvious methods for improving usability. The previous usability evaluations of the Tor system brought users into the fold and attempted to find the most common problems being faced by users of the system from their perspective, this was an important step in improving the accessibility of the system. Improvements stemming from this research typically involved tidying the system of connecting to Tor, removing jargon words, and removing complex configuration methods from users' view. These studies have not, however, identified more complex usability issues which are not easily identified by users. All the projects of this thesis, however, focused on more niche usability improvements addressing issues which were not likely to be identified as problems by users, but which still addressed critical areas where errors could occur. SSL stripping attacks, for example, can be devastating to a user's anonymity but these attacks have thus far not been dealt with from a usability perspective. A large portion of the work in this thesis deals with SSL Stripping attacks and enhances usability to reduce their likelihood of success. Similarly, work completed in this thesis on delay in high-latency systems evaluates users' success at tasks as delay varies. Crucially, its results are based on the direct measure of task completion times, and not on users' self-reported satisfaction. Traditional usability evaluations of delay in anonymity systems acquire user feedback on if they feel the current level of delay in a system is tolerable, my work in this thesis went beyond this and used direct measures for many delay values. This achieved a more nuanced view of delay in anonymity systems, which, once again would not have been produced by a typical user-centred methodological approach.

Overall, this thesis demonstrates alongside traditional usability approaches to anonymity systems, broader usability improvements that are not motivated by user reports can be essential and provide critical safety improvements to the anonymity system.

7.2 Work Done in this Thesis

Usability in anonymity systems can mean many things, and the chapters of my thesis discuss several different types of usability in anonymity. The foundational usability issue in anonymity systems is latency; most anonymity systems introduce latency, either by design or due to undesired overheads. Latency is an obvious usability issue as, depending on the task, users can become very frustrated if their interactions with software are delayed.

As latency is crucial factor of usability in anonymity, Chapter 3 showed an investigation into tolerable latency levels for high-latency mixnet systems. Latency is the biggest usability hurdle for high-latency anonymity systems. Although advances have been made in making more efficient high-latency systems like the Loopix mixnet, no real usability research has been done into these modern systems. The study in this section of the thesis made a first attempt to evaluate what kinds of usability would be tolerable in a modern mixnet system.

Latency has long been the most important factor in making systems more usable and does remain a problem even for the highly mature Tor system [47]. Despite this, there is a wide array of other usability issues that have become apparent over the years. While the underlying technology of anonymity systems is highly effective, one of the largest threats to users' anonymity is due to the content of data sent through the system. This could be an accidental disclosure by the user or through nefarious tracking systems. The Tor browser attempts to mitigate most types of tracking; however, configuring anonymity software to prevent tracking and other anonymity failures is an ongoing development. One way in which anonymity can break down is through man-in-the-middle attacks. Malicious Tor exit node operators can trivially intercept users' content and either alter or snoop on any user activity. Only encrypted and authenticated communication can defeat this type of attack.

Defeating man-in-the-middle-attacks has become a usability problem through the deployment of HTTPS-Only modes as a mechanism to enforce encrypted and au-

thenticated communication over Tor. Chapter 4 discussed my project, which took a first look at how these HTTPS-Only modes should function in Tor Browser, and in particular, what kind of information should be given to users in the warning page displayed. The chapter gave a clear outline of the current state of HTTPS-Only modes in other browsers before exploring my detailed qualitative survey of Tor experts. In the survey, participants were asked about their views on risk to Tor users. Questions included asking if the risk faced from non-HTTPS web pages was different for Tor users and what technical aspects users need to understand in order to make informed decisions. Participants were then presented with a sample of three current warning pages and asked to evaluate them based on their suitability for Tor Browser. The project produced a wide range of themes that can direct feature research and future design of HTTPS-Only mode warning pages, not just for Tor, but for other browsers too.

The next Chapter, Chapter 5, produces three new warning pages that were crafted explicitly from the themes identified in the previous chapter. The three new warning pages attempted to allow users to make more informed decisions based on three different heuristics. Firstly, the popularity of the webpage; more popular websites tend to implement HTTPS and therefore, a warning appearing is more likely to be suspicious. Secondly, the context that the page is being visited in, i.e. will the user be entering personal data? Finally, a Tor-specific warning was devised, which warned about the extra risk that is faced by malicious exit nodes in Tor. To evaluate these new pages, a set of criteria was developed describing under what conditions a user should or should not proceed to websites without HTTPS. The large-scale evaluation study that was then performed showed that the population warning page was statistically significantly improved upon the current Firefox warning page. The results showed how the warning pages affected users' choices in different browsing scenarios, which will inform the future design of improved HTTPS-Only mode warning pages.

The final piece of work presented in the thesis addressed another aspect of usability.

While most changes in Tor Browser provide additional protections on top of regular browsers, some changes are features that are removed to increase privacy protections. Chapter 6 addresses one of these changes. HSTS is a feature that provides protection to some extent from man-in-the-middle attacks during web browsing, but it has been disabled in Tor Browser due to its tracking potential. The new protocol presented in this chapter allows HSTS data to be crowdsourced in a privacy-preserving way. Tor Browser users can send HSTS to a centralised database without revealing which websites they have been browsing, whilst other Tor Browser users can make use of this data to protect themselves from man-in-the-middle attacks like SSL Stripping. The mechanism can prevent users from becoming victims of these types of attacks and thus increases the strength of their anonymity protection by removing a common mechanism for deanonymisation.

The research I completed in these four projects contributes to creating anonymity network software that is safer and more usable in a number of ways. The thesis shows how usability can manifest in a variety of different ways, like latency issues, warning page design, and man-in-the-middle protections, and it proposes practical and tested improvements in all of these areas.

7.3 Future Work

Much of the work in this thesis can be framed as explorations into new topics which have not been researched thoroughly. All four of the projects presented raise new questions, and lay the groundwork for further work which could improve usability across high and low-latency anonymity systems.

The work of Chapter 3 was a preliminary look into how the latency of the most modern high-latency anonymity systems affects usability. The experiment conducted here was a general one, aiming to generalise as much as possible to a wide variety of applications that could be used. An important point that was discussed is that the tolerance of users to additional latency is widely dependent on the application. Future work on this topic could include measuring the tolerance of users in more

specific types of applications.

Chapters 4 and 5 both pertain to HTTPS-Only mode warning pages, a relatively new development, and one which has not been studied in the literature. The first steps taken in these two chapters to understand HTTPS-Only mode warning pages are the first look into a feature which is likely to become much more popular in the future as HTTPS adoption increases. The two surveys conducted did produce a single new warning page that was significantly improved upon the current warning page, however, the studies also produced more general guidelines which can inform future designs. The improvements that were made here are clearly only a partial improvement and further work using the data I collected will improve these warnings further.

The protocol presented in Chapter 6 is a mechanism that allows the use of HSTS in Tor Browser. The probabilistic nature of the protocol means that there is an inherent false positive and false negative rate which was measured in the simulations described. Future work should study how the protocol will evolve as the rate of HTTPS adoption increases. Increasing HTTPS adoption on the internet will increase the utility of the protocol further and reduce the level of false positives. Secondly, the protocol presents new warnings to users which have not been studied. Much like in the previous two chapters, these new warnings, as well as the protocol as a whole, should be tested for usability. It is not yet clear how users will react to such a protocol, and what the best way to present this protocol is so that users will obtain the most utility from it. Answering these usability questions would enhance the protocol and thus give the opportunity to further improve the usability of Tor Browser.

The projects in this thesis outline a number of new avenues for usability improvement for both high and low-latency systems, which, if followed, can continue the large body of work on improving the usability of anonymity systems overall and allowing users to obtain anonymity on the internet more easily and more effectively.

Bibliography

[1] Roger Dingledine and Nick Mathewson. “Anonymity Loves Company: Usability and the Network Eect”. In: *WEIS* (2006).

[2] *The Tor Project*. 2023. URL: `https://torproject.org` (visited on 08/15/2022).

[3] Jeff Hodges, Collin Jackson, and Adam Barth. *HTTP Strict Transport Security (HSTS)*. Request for Comments. Internet Engineering Task Force, Nov. 2012. URL: `https://datatracker.ietf.org/doc/rfc6797`.

[4] Andreas Pfitzmann and Marit Köhntopp. “Anonymity, Unobservability, and Pseudonymity A Proposal for Terminology”. In: *Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability*. 2001. URL: `https://doi.org/10.1007/3-540-44702-4_1`.

[5] Steven J. Murdoch. “Quantifying and Measuring Anonymity”. In: *Data Privacy Management and Autonomous Spontaneous Security*. 2014. URL: `http://link.springer.com/10.1007/978-3-642-54568-9_1`.

[6] Sajin Sasy and Ian Goldberg. “SoK: Metadata-Protecting Communication Systems”. In: *Proceedings on Privacy Enhancing Technologies* (2024). URL: `https://petsymposium.org/popets/2024/popets-2024-0030.php`.

[7] David Chaum. “The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability”. In: *Journal of Cryptology* (1988). URL: http://link.springer.com/10.1007/BF00206326.

[8] David L. Chaum. “Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms”. In: *Communications of the ACM* (Feb. 1, 1981). URL: https://doi.org/10.1145/358549.358563.

[9] *Mixmaster Protocol Version 2*. 2004. URL: https://www.mixmin.net/draft-sassaman-mixmaster-XX.html (visited on 04/15/2022).

[10] Andrei Serjantov, Roger Dingledine, and Paul Syverson. “From a Trickle to a Flood: Active Attacks on Several Mix Types”. In: *Information Hiding*. 2003. URL: http://link.springer.com/10.1007/3-540-36415-3_3.

[11] Dogan Kesdogan, Jan Egner, and Roland Büschkes. “Stop- and-Go-MIXes Providing Probabilistic Anonymity in an Open System”. In: *Information Hiding*. 1998. URL: http://link.springer.com/10.1007/3-540-49380-8_7.

[12] Oliver Berthold and Heinrich Langos. “Dummy Traffic against Long Term Intersection Attacks”. In: *Privacy Enhancing Technologies*. 2003. URL: http://link.springer.com/10.1007/3-540-36467-6_9.

[13] Ania M. Piotrowska, Jamie Hayes, Tariq Elahi, Sebastian Meiser, and George Danezis. “The Loopix Anonymity System”. In: *26th USENIX Security Symposium (USENIX Security 17)*. 2017. URL: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/piotrowska.

[14] Daniel Hugenroth, Martin Kleppmann, and Alastair R. Beresford. “Rollercoaster: An Efficient Group-Multicast Scheme for Mix Networks”. In: *30th USENIX Security Symposium (USENIX Security 21)*. 2021. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/hugenroth.

[15] Debajyoti Das, Sebastian Meiser, Esfandiar Mohammadi, and Aniket Kate. "Divide and Funnel: A Scaling Technique for Mix-Networks". In: *IEEE 37th Computer Security Foundations Symposium (CSF)*. July 2024. URL: https://ieeexplore.ieee.org/abstract/document/10664379.

[16] Claudia Diaz, Carmela Troncoso, and Andrei Serjantov. "On the Impact of Social Network Profiling on Anonymity". In: *Privacy Enhancing Technologies*. 2008. URL: http://link.springer.com/10.1007/978-3-540-70630-4_4.

[17] Roger Dingledine, Nick Mathewson, and Paul Syverson. "Tor: The Second-Generation Onion Router". In: *13th USENIX Security Symposium (USENIX Security 04)*. 2004. URL: https://www.usenix.org/conference/13th-usenix-security-symposium/tor-second-generation-onion-router.

[18] David M. Goldschlag, Michael G. Reed, and Paul F. Syverson. "Hiding Routing Information". In: *Information Hiding*. 1996. URL: http://link.springer.com/10.1007/3-540-61996-8_37.

[19] Markus Huber, Martin Mulazzani, and Edgar Weippl. "Tor HTTP Usage and Information Leakage". In: *Communications and Multimedia Security*. 2010.

[20] Timothy G. Abbott, Katherine J. Lai, Michael R. Lieberman, and Eric C. Price. "Browser-Based Attacks on Tor". In: *Privacy Enhancing Technologies*. 2007.

[21] *Explanation of a Potential Technique to Deanonymise Users of the TOR Network*. URL: https://www.aclu.org/wp-content/uploads/legal-documents/Explanation%20of%20a%20Potential%20Technique%20to%20Deanonymise%20Users%20of%20the%20TOR%20Network.pdf (visited on 11/02/2023).

[22] Mike Perry. *Holy Shit I Caught 1*. Aug. 28, 2006. URL: https://lists.torproject.org/pipermail/tor-talk/2006-August/001766.html (visited on 11/02/2023).

[23] *DEranged Security: "Time to Reveal"* Oct. 20, 2007. URL: https://web.archive.org/web/20071020181404/http://www.derangedsecurity.com/time-to-reveal%E2%80%A6/ (visited on 11/02/2023).

[24] *The Case of The Modified Binaries*. Leviathan Security Group. URL: https://www.leviathansecurity.com/media/the-case-of-the-modified-binaries (visited on 11/02/2023).

[25] FSLabs. *OnionDuke: APT Attacks Via the Tor Network*. Nov. 14, 2014. URL: https://archive.f-secure.com/weblog/archives/00002764.html (visited on 11/02/2023).

[26] Sambuddho Chakravarty, Georgios Portokalidis, Michalis Polychronakis, and Angelos D. Keromytis. “Detecting Traffic Snooping in Tor Using Decoys”. In: *Recent Advances in Intrusion Detection*. 2011.

[27] Sambuddho Chakravarty, Georgios Portokalidis, Michalis Polychronakis, and Angelos D. Keromytis. “Detection and Analysis of Eavesdropping in Anonymous Communication Networks”. In: *International Journal of Information Security* (June 1, 2015). URL: https://doi.org/10.1007/s10207-014-0256-7.

[28] Philipp Winter, Richard Köwer, Martin Mulazzani, Markus Huber, Sebastian Schrittwieser, Stefan Lindskog, and Edgar Weippl. “Spoiled Onions: Exposing Malicious Tor Exit Relays”. In: *Privacy Enhancing Technologies*. 2014.

[29] Amirali Sanatinia and Guevara Noubir. “Honey Onions: A Framework for Characterizing and Identifying Misbehaving Tor HSDirs”. In: *2016 IEEE*

Conference on Communications and Network Security (CNS). Oct. 2016. URL: http://ieeexplore.ieee.org/document/7860478/.

[30] nusenu. *The Growing Problem of Malicious Relays on the Tor Network*. Medium. Aug. 2, 2020. URL: https://nusenu.medium.com/the-growing-problem-of-malicious-relays-on-the-tor-network-2f14198af548 (visited on 11/03/2023).

[31] nusenu. *How Malicious Tor Relays Are Exploiting Users in 2020 (Part I)*. Medium. Dec. 31, 2020. URL: https://nusenu.medium.com/how-malicious-tor-relays-are-exploiting-users-in-2020-part-i-1097575c0cac (visited on 02/21/2023).

[32] nusenu. *Tracking One Year of Malicious Tor Exit Relay Activities (Part II)*. Medium. May 8, 2021. URL: https://nusenu.medium.com/tracking-one-year-of-malicious-tor-exit-relay-activities-part-ii-85c80875c5df (visited on 11/03/2023).

[33] Isabela. *Tor Security Advisory: Exit Relays Running Sslstrip in May and June 2020*. Aug. 14, 2020. URL: https://blog.torproject.org/bad-exit-relays-may-june-2020/ (visited on 01/23/2023).

[34] *Tor Browser: A Legacy of Advancing Private Browsing Innovation*. URL: https://blog.torproject.org/tor-browser-advancing-privacy-innovation/ (visited on 11/21/2023).

[35] Mike Perry. *To Toggle, or Not to Toggle: The End of Torbutton*. URL: https://blog.torproject.org/toggle-or-not-toggle-end-torbutton/ (visited on 11/21/2023).

[36] Mike Perry. *Tor Browser Bundle 3.5 Is Released*. URL: https://blog.torproject.org/tor-browser-bundle-35-released/ (visited on 09/07/2023).

[37] Antonela Debiasi. *New Release: Tor Browser 10.5*. URL: https://blog.torproject.org/new-release-tor-browser-105/ (visited on 12/05/2023).

[38] Mike Perry, Erin Clark, Steven Murdoch, and Georg Koppen. *The Design and Implementation of the Tor Browser [DRAFT]*. June 15, 2018. URL: https://2019.www.torproject.org/projects/torbrowser/design/ (visited on 02/21/2023).

[39] Ben Collier. *Tor*. 2024.

[40] *Tor User Research Reports*. URL: https://community.torproject.org/user-research/reports/ (visited on 11/21/2023).

[41] Tor Project. *Tor Browser User Survey Report*. GitLab. Nov. 16, 2021. URL: https://gitlab.torproject.org/tpo/ux/research/-/blob/master/reports/2021/tor-browser-user-survey-public.pdf (visited on 11/21/2023).

[42] Vidalia Project. *Vidalia - Home*. Feb. 6, 2009. URL: https://web.archive.org/web/20090206223846/https://vidalia-project.net/ (visited on 11/15/2023).

[43] Jeremy Clark, P. C. van Oorschot, and Carlisle Adams. "Usability of Anonymous Web Browsing: An Examination of Tor Interfaces and Deployability". In: *Proceedings of the 3rd Symposium on Usable Privacy and Security*. July 18, 2007. URL: https://dl.acm.org/doi/10.1145/1280680.1280687.

[44] Greg Norcie, Kelly Caine, and L Jean Camp. "Eliminating Stop-Points in the Installation and Use of Anonymity Systems: A Usability Evaluation of the Tor Browser Bundle". In: *5th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETS)* (2012).

[45] Greg Norcie, Jim Blythe, Kelly Caine, and L. Jean Camp. "Why Johnny Can't Blow the Whistle: Identifying and Reducing Usability Issues in Anonymity Systems". In: *Proceedings 2014 Workshop on Usable Security*. 2014. URL: https://www.ndss-symposium.org/ndss2014/workshop-usable-security-usec-2014-programme/why-johnny-cant-blow-whistle-identifying-and-reducing-usability-issues-anonymity-systems.

[46] Kevin Gallagher, Sameer Patil, and Nasir Memon. "New Me: Understanding Expert and Non-Expert Perceptions and Usage of the Tor Anonymity Network". In: *Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017)*. 2017. URL: https://www.usenix.org/conference/soups2017/technical-sessions/presentation/gallagher.

[47] Akshaya Kumar Viswanathan, Andreas Hulsing, and Katharina Kohls. "A Method to Evaluate the Usability Issues Experienced When Using Tor Browser". Eindhoven University of Technology, 2022.

[48] Kevin Gallagher, Sameer Patil, Brendan Dolan-Gavitt, Damon McCoy, and Nasir Memon. "Peeling the Onion's User Experience Layer: Examining Naturalistic Use of the Tor Browser". In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. Oct. 15, 2018. URL: https://dl.acm.org/doi/10.1145/3243734.3243803.

[49] Linda Lee, David Fifield, Nathan Malkin, Ganesh Iyer, Serge Egelman, and David Wagner. "A Usability Evaluation of Tor Launcher". In: *Proceedings on Privacy Enhancing Technologies* (July 1, 2017). URL: https://petsymposium.org/popets/2017/popets-2017-0030.php.

[50] *Improving the User Experience of Connecting to Tor in Tor Browser 10.5 | Tor Project*. URL: https://blog.torproject.org/improving-ux-connecting-to-tor-105/ (visited on 09/07/2023).

[51] Benjamin Fabian, Florian Goertz, Steffen Kunz, Sebastian Müller, and Mathias Nitzsche. “Privately Waiting A Usability Analysis of the Tor Anonymity Network”. In: *Sustainable E-Business Management*. 2010.

[52] Sebastian Müller, Franziska Brecht, Benjamin Fabian, Steffen Kunz, and Dominik Kunze. “Distributed Performance Measurement and Usability Assessment of the Tor Anonymization Network”. In: *Future Internet* (2 June 2012). URL: https://www.mdpi.com/1999-5903/4/2/488.

[53] *New Release: Tor Browser 12.5*. URL: https://blog.torproject.org/new-release-tor-browser-125/ (visited on 12/06/2023).

[54] Matthew Finkel. *New Release: Tor Browser 9.0.7*. Mar. 23, 2020. URL: https://blog.torproject.org/new-release-tor-browser-907/ (visited on 11/14/2023).

[55] *New Release: Tor Browser 9.0*. URL: https://blog.torproject.org/new-release-tor-browser-90/ (visited on 12/06/2023).

[56] *New Release: Tor Browser 8.5*. URL: https://blog.torproject.org/new-release-tor-browser-85/ (visited on 12/06/2023).

[57] Mike Perry. *HTTPS Everywhere Firefox Addon Helps You Encrypt Web Traffic*. June 18, 2010. URL: https://blog.torproject.org/https-everywhere-firefox-addon-helps-you-encrypt-web-traffic/ (visited on 11/03/2023).

[58] Alexis Hancock and Bill Budington. *How HTTPS Everywhere Keeps Protecting Users On An Increasingly Encrypted Web*. Electronic Frontier Foundation. Dec. 13, 2018. URL: https://www.eff.org/deeplinks/2018/12/how-https-everywhere-keeps-protecting-users-increasingly-encrypted-web (visited on 11/02/2023).

[59] EFF. *Https-Everywhere Changelog*. URL: https://www.eff.org/files/Changelog.txt (visited on 11/06/2023).

[60] Suphannee Sivakorn, Angelos D. Keromytis, and Jason Polakis. "That's the Way the Cookie Crumbles: Evaluating HTTPS Enforcing Mechanisms". In: *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*. Oct. 24, 2016. URL: https://dl.acm.org/doi/10.1145/2994620.2994638.

[61] Wilfried Mayer and Martin Schmiedecker. "TLScompare: Crowdsourcing Rules for HTTPS Everywhere". In: *Proceedings of the 25th International Conference Companion on World Wide Web*. Apr. 11, 2016. URL: https://dl.acm.org/doi/10.1145/2872518.2888606.

[62] Collin Jackson and Adam Barth. "Forcehttps: Protecting High-Security Web Sites from Network Attacks". In: *Proceeding of the 17th International Conference on World Wide Web - WWW '08*. 2008. URL: http://portal.acm.org/citation.cfm?doid=1367497.1367569.

[63] Chromium. *HSTS Preload List Submission*. 2022. URL: https://hstspreload.org/ (visited on 08/15/2022).

[64] J. V. Roig and Eunice Grace Gatdula. *HSTS Preloading Is Ineffective as a Long-Term, Wide-Scale MITM-Prevention Solution: Results from Analyzing the 2013 - 2017 HSTS Preload List*. May 10, 2019. arXiv: 1905.04436 [cs]. URL: http://arxiv.org/abs/1905.04436 (visited on 08/16/2022). Pre-published.

[65] Majestic Analytics. *Majestic Million*. 2022. URL: https://majestic.com/reports/majestic-million (visited on 05/18/2022).

[66] Mozilla. *Web Security*. July 2018. URL: https://infosec.mozilla.org/guidelines/web_security (visited on 02/21/2023).

[67] Moxie Marlinspike. “Some Tricks for Defeating SSL in Practice”. 2009. URL: http://2015.hack.lu/archive/2009/moxie-marlinspike-some_tricks_for_defeating_ssl_in_practice.pdf.

[68] Moxie Marlinspike. *Moxie0/Sslstrip*. Oct. 31, 2023. URL: https://github.com/moxie0/sslstrip.

[69] Leonardo Nve. *SSLStrip+*. Oct. 18, 2023. URL: https://github.com/LeonardoNve/sslstrip2.

[70] *Think about Using DNS over HTTPS for Tor Browser*. June 4, 2019. URL: https://gitlab.torproject.org/tpo/applications/tor-browser/-/issues/30753 (visited on 11/21/2023).

[71] Nate Nelson and Dark Reading September 29. *Spyware Vendor Targets Egyptian Orgs With Rare iOS Exploit Chain*. Dark Reading. Sept. 29, 2023. URL: https://www.darkreading.com/dr-global/spyware-vendor-egyptian-orgs-ios-exploit-chain (visited on 10/03/2023).

[72] Shuo Chen, Ziqing Mao, Yi-Min Wang, and Ming Zhang. “Pretty-Bad-Proxy: An Overlooked Adversary in Browsers’ HTTPS Deployments”. In: *2009 30th IEEE Symposium on Security and Privacy*. May 2009. URL: http://ieeexplore.ieee.org/document/5207655/.

[73] Nick Nikiforakis, Yves Younan, and Wouter Joosen. “HProxy: Client-Side Detection of SSL Stripping Attacks”. In: *Detection of Intrusions and Malware, and Vulnerability Assessment*. 2010.

[74] Suphannee Sivakorn, Iasonas Polakis, and Angelos D. Keromytis. “The Cracked Cookie Jar: HTTP Cookie Hijacking and the Exposure of Private Information”. In: *2016 IEEE Symposium on Security and Privacy (SP)*. May 2016. URL: https://ieeexplore.ieee.org/document/7546532.

[75] Taejoong Chung, David Choffnes, and Alan Mislove. “Tunneling for Transparency: A Large-Scale Analysis of End-to-End Violations in the Internet”. In: *Proceedings of the 2016 Internet Measurement Conference*.

Nov. 14, 2016. URL: https://dl.acm.org/doi/10.1145/2987443.2987455.

[76] Abner Mendoza, Phakpoom Chinprutthiwong, and Guofei Gu. "Uncovering HTTP Header Inconsistencies and the Impact on Desktop/Mobile Websites". In: *Proceedings of the 2018 World Wide Web Conference*. Apr. 23, 2018. URL: https://dl.acm.org/doi/10.1145/3178876.3186091.

[77] Mingming Zhang, Xiaofeng Zheng, Kaiwen Shen, Ziqiao Kong, Chaoyi Lu, Yu Wang, Haixin Duan, Shuang Hao, Baojun Liu, and Min Yang. "Talking with Familiar Strangers: An Empirical Study on HTTPS Context Confusion Attacks". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. Oct. 30, 2020. URL: https://dl.acm.org/doi/10.1145/3372297.3417252.

[78] Vyron Kampourakis, Georgios Kambourakis, Efstratios Chatzoglou, and Christos Zaroliagis. "Revisiting Man-in-the-Middle Attacks against HTTPS". In: *Network Security* (Mar. 2022). URL: https://www.magonlinelibrary.com/doi/full/10.12968/S1353-4858%2822%2970028-1.

[79] *Intent to Ship: HTTPS Upgrades*. URL: https://groups.google.com/a/chromium.org/g/blink-dev/c/cAS525en8XE#:~:text=We%20enabled%20HTTPS,%EE%97%93 (visited on 11/07/2023).

[80] Jose Selvi. *Bypassing HTTP Strict Transport Security*. INCIDE, 2014. URL: https://www.blackhat.com/docs/eu-14/materials/eu-14-Selvi-Bypassing-HTTP-Strict-Transport-Security-wp.pdf.

[81] Sheila Berta and Sergio De los Santos. "Breaking Out HSTS (and HPKP) On Firefox, IE/Edge and (Possibly) Chrome". Dec. 7, 2017. URL: https://www.youtube.com/watch?v=dPnU9_pXJ5k.

[82] Andrew Christensen. *Practical Onion Hacking*. FortConsult, 2006. URL: https://dl.packetstormsecurity.net/0610-advisories/Practical_Onion_Hacking.pdf.

[83] Adrienne Porter Felt, Robert W. Reeder, Alex Ainslie, Helen Harris, Max Walker, Christopher Thompson, Mustafa Embre Acer, Elisabeth Morant, and Sunny Consolvo. "Rethinking Connection Security Indicators". In: *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*. 2016. URL: https://www.usenix.org/conference/soups2016/technical-sessions/presentation/porter-felt.

[84] Lydia Kraus, Martin Ukrop, Vashek Matyas, and Tobias Fiebig. "Evolution of SSL/TLS Indicators and Warnings in Web Browsers". In: *Security Protocols XXVII*. 2020.

[85] Christopher Thompson, Martin Shelton, Emily Stark, Maximilian Walker, Emily Schechter, and Adrienne Porter Felt. "The Web's Identity Crisis: Understanding the Effectiveness of Website Identity Indicators". In: *28th USENIX Security Symposium (USENIX Security 19)*. 2019.

[86] Rachna Dhamija, J. D. Tygar, and Marti Hearst. "Why Phishing Works". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Apr. 22, 2006. URL: https://dl.acm.org/doi/10.1145/1124772.1124861.

[87] Tara Whalen and Kori M. Inkpen. "Gathering Evidence: Use of Visual Security Cues in Web Browsers". In: *Proceedings of Graphics Interface 2005*. May 7, 2005.

[88] Stuart E. Schechter, Rachna Dhamija, Andy Ozment, and Ian Fischer. "The Emperor's New Security Indicators". In: *2007 IEEE Symposium on Security and Privacy (SP '07)*. May 2007.

[89] kexugit. *Mixed Content and Internet Explorer 8.0*. May 14, 2009. URL: https://learn.microsoft.com/en-us/archive/blogs/askie/mixed-content-and-internet-explorer-8-0 (visited on 11/21/2023).

[90] Chris Evans and Tom Sepez. *Trying to end mixed scripting vulnerabilities*. Google Online Security Blog. URL: https://security.googleblog.com/2011/06/trying-to-end-mixed-scripting.html (visited on 11/21/2023).

[91] Lucas Garron and Chris Palmer. *Simplifying the Page Security Icon in Chrome*. Google Online Security Blog. URL: https://security.googleblog.com/2015/10/simplifying-page-security-icon-in-chrome.html (visited on 11/21/2023).

[92] *Moving towards a More Secure Web*. Google Online Security Blog. URL: https://security.googleblog.com/2016/09/moving-towards-more-secure-web.html (visited on 11/21/2023).

[93] Emily Schechter. *Next Steps Toward More Connection Security*. Google Online Security Blog. URL: https://security.googleblog.com/2017/04/next-steps-toward-more-connection.html (visited on 11/21/2023).

[94] Emily Schechter. *A Secure Web Is Here to Stay*. Google Online Security Blog. URL: https://security.googleblog.com/2018/02/a-secure-web-is-here-to-stay.html (visited on 11/21/2023).

[95] Shweta Panditrao. *Protecting Google Chrome Users from Insecure Forms*. Chromium Blog. URL: https://blog.chromium.org/2020/08/protecting-google-chrome-users-from.html (visited on 11/21/2023).

[96] Shweta Panditrao, Devon O'Brien, and Emily Stark. *Increasing HTTPS Adoption*. Chromium Blog. 2021. URL: https:

//blog.chromium.org/2021/07/increasing-https-adoption.html (visited on 08/12/2022).

[97] Dongwan Shin and Rodrigo Lopes. "An Empirical Study of Visual Security Cues to Prevent the SSLstripping Attack". In: *Proceedings of the 27th Annual Computer Security Applications Conference*. Dec. 5, 2011. URL: https://dl.acm.org/doi/10.1145/2076732.2076773.

[98] Kat Krol, Matthew Moroz, and M. Angela Sasse. "Don't Work. Can't Work? Why It's Time to Rethink Security Warnings". In: *2012 7th International Conference on Risks and Security of Internet and Systems (CRiSIS)*. Oct. 2012.

[99] Bonnie Anderson, Anthony Vance, Brock Kirwan, David Eargle, and Seth Howard. "Users Arent (Necessarily) Lazy: Using NeuroIS to Explain Habituation to Security Warnings". In: *ICIS 2014 Proceedings* (Dec. 15, 2014). URL: https://aisel.aisnet.org/icis2014/proceedings/ISSecurity/28.

[100] Heather Molyneaux, Irina Kondratova, and Elizabeth Stobert. "Understanding Perceptions: User Responses to Browser Warning Messages". In: *HCI for Cybersecurity, Privacy and Trust*. 2019.

[101] Bonnie Brinton Anderson, C. Brock Kirwan, Jeffrey L. Jenkins, David Eargle, Seth Howard, and Anthony Vance. "How Polymorphic Warnings Reduce Habituation in the Brain: Insights from an fMRI Study". In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. Apr. 18, 2015. URL: https://dl.acm.org/doi/10.1145/2702123.2702322.

[102] José Carlos Brustoloni and Ricardo Villamarín-Salomón. "Improving Security Decisions with Polymorphic and Audited Dialogs". In: *Proceedings of the 3rd Symposium on Usable Privacy and Security*. July 18, 2007. URL: https://dl.acm.org/doi/10.1145/1280680.1280691.

[103] Christoph Kerschbaumer, Julian Gaibler, Arthur Edelstein, and Thyla van der Merwe. *Firefox 83 Introduces HTTPS-Only Mode*. Mozilla Security Blog. Nov. 17, 2020. URL: https://blog.mozilla.org/security/2020/11/17/firefox-83-introduces-https-only-mode (visited on 08/02/2023).

[104] Chris Thompson. *Enable HTTPS-First Mode Flag by Default*. Chromium Source. Aug. 9, 2021. URL: https://chromium.googlesource.com/chromium/src/+/4f812999d6680d77ab368f4fbad1338359bb6937 (visited on 02/03/2024).

[105] Microsoft Edge Blog. *Available for Preview: Automatic HTTPS Helps Keep Your Browsing More Secure*. Microsoft Edge Blog. June 1, 2021. URL: https://blogs.windows.com/msedgedev/2021/06/01/available-for-preview-automatic-https-helps-keep-your-browsing-more-secure/ (visited on 08/12/2022).

[106] Jen Simmons. *New WebKit Features in Safari 15*. WebKit. Oct. 26, 2021. URL: https://webkit.org/blog/11989/new-webkit-features-in-safari-15/ (visited on 02/03/2024).

[107] Brave Privacy Team. *Brave: HTTPS by Default*. Brave. Feb. 9, 2023. URL: https://brave.com/privacy-updates/22-https-by-default/ (visited on 02/03/2024).

[108] Q-Success. *Historical Yearly Trends in the Usage Statistics of Site Elements for Websites, January 2024*. Historical yearly trends in the usage statistics of site elements for websites. URL: https://w3techs.com/technologies/history_overview/site_element/all/y (visited on 01/01/2024).

[109] Katharina Krombholz, Karoline Busse, Katharina Pfeffer, Matthew Smith, and Emanuel von Zezschwitz. ""If HTTPS Were Secure, I Wouldn't Need

2FA" - End User and Administrator Mental Models of HTTPS". In: *2019 IEEE Symposium on Security and Privacy (SP)*. May 2019.

[110] *Disable Plaintext HTTP Clearnet Connections*. GitLab. Aug. 7, 2016. URL: https://gitlab.torproject.org/tpo/applications/tor-browser/-/issues/19850 (visited on 11/03/2023).

[111] Duncan Russell. *New Release: Tor Browser 11.5 | Tor Project*. July 14, 2022. URL: https://blog.torproject.org/new-release-tor-browser-115/ (visited on 11/03/2023).

[112] Eric Butler. *Firesheep*. Firesheep. 2010. URL: https://codebutler.com/2010/10/24/firesheep/ (visited on 08/24/2023).

[113] Muhammad Talha Paracha, Balakrishnan Chandrasekaran, David Choffnes, and Dave Levin. "A Deeper Look at Web Content Availability and Consistency over HTTP/S". In: *Network Traffic Measurement and Analysis Conference* (2020).

[114] W3Techs. *Usage Statistics of Default Protocol Https for Websites, October 2023*. Oct. 2023. URL: https://w3techs.com/technologies/details/ce-httpsdefault (visited on 10/17/2023).

[115] *HTTPS Encryption on the Web Google Transparency Report*. URL: https://transparencyreport.google.com/https/overview?hl=en_GB (visited on 08/09/2023).

[116] Adrienne Porter Felt, Richard Barnes, April King, Chris Palmer, Chris Bentzel, and Parisa Tabriz. "Measuring HTTPS Adoption on the Web". In: *26th USENIX Security Symposium (USENIX Security 17)*. 2017. URL: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/felt.

[117] Peter Eckersley. *Launching in 2015: A Certificate Authority to Encrypt the Entire Web*. Electronic Frontier Foundation. Nov. 18, 2014. URL: https://www.eff.org/deeplinks/2014/11/certificate-authority-encrypt-entire-web (visited on 10/20/2023).

[118] *Let's Encrypt*. URL: https://letsencrypt.org/ (visited on 12/01/2023).

[119] Josh Aas et al. "Let's Encrypt: An Automated Certificate Authority to Encrypt the Entire Web". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. Nov. 6, 2019. URL: https://dl.acm.org/doi/10.1145/3319535.3363192.

[120] Matthew Bernhard, Jonathan Sharman, Claudia Ziegler Acemyan, Philip Kortum, Dan S. Wallach, and J. Alex Halderman. "On the Usability of HTTPS Deployment". In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. May 2, 2019. URL: https://dl.acm.org/doi/10.1145/3290605.3300540.

[121] Troy Hunt. *HTTPS Adoption Has Reached the Tipping Point*. Troy Hunt. Jan. 30, 2017. URL: https://www.troyhunt.com/https-adoption-has-reached-the-tipping-point/ (visited on 09/12/2023).

[122] Ariana Mirian, Christopher Thompson, Stefan Savage, Geoffrey M. Voelker, and Adrienne Porter Felt. *HTTPS Adoption in the Longtail*. Google and UC San Diego, 2018.

[123] G. Danezis, R. Dingledine, and N. Mathewson. "Mixminion: Design of a Type III Anonymous Remailer Protocol". In: *2003 Symposium on Security and Privacy, 2003.* May 2003.

[124] Andrei Serjantov and Steven J. Murdoch. "Message Splitting Against the Partial Adversary". In: *Privacy Enhancing Technologies*. 2006.

[125] Claudia Diaz. "Mix Networks". In: *Encyclopedia of Cryptography, Security and Privacy*. 2019. URL: https://doi.org/10.1007/978-3-642-27739-9_1754-1.

[126] Claudia-Lavinia Ignat, Gérald Oster, Olivia Fox, Valerie L. Shalin, and François Charoy. “How Do User Groups Cope with Delay in Real-Time Collaborative Note Taking”. In: *ECSCW 2015: Proceedings of the 14th European Conference on Computer Supported Cooperative Work, 19-23 September 2015, Oslo, Norway*. 2015.

[127] Gary M. Olson and Judith S. Olson. “Distance Matters”. In: *Human-Computer Interaction* (Sept. 1, 2000). URL: https://doi.org/10.1207/S15327051HCI1523_4.

[128] Stefan Köpsell. “Low Latency Anonymous Communication How Long Are Users Willing to Wait?” In: *Emerging Trends in Information and Communication Security*. 2006.

[129] Quang-Vinh Dang and Claudia-Lavinia Ignat. “Performance of Real-Time Collaborative Editors at Large Scale: User Perspective”. In: *2016 IFIP Networking Conference (IFIP Networking) and Workshops*. May 2016. URL: http://ieeexplore.ieee.org/document/7497258/.

[130] Xiao Bai, Ioannis Arapakis, B. Barla Cambazoglu, and Ana Freire. “Understanding and Leveraging the Impact of Response Latency on User Behaviour in Web Search”. In: *ACM Transactions on Information Systems* (Apr. 30, 2018). URL: https://dl.acm.org/doi/10.1145/3106372.

[131] Ioannis Arapakis, Souneil Park, and Martin Pielot. “Impact of Response Latency on User Behaviour in Mobile Web Search”. In: *Proceedings of the 2021 Conference on Human Information Interaction and Retrieval*. Mar. 14, 2021. URL: https://dl.acm.org/doi/10.1145/3406522.3446038.

[132] Darren Gergle, Robert E. Kraut, and Susan R. Fussell. “The Impact of Delayed Visual Feedback on Collaborative Performance”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Apr. 22, 2006. URL: https://dl.acm.org/doi/10.1145/1124772.1124968.

[133] Fiona Nah. “A Study on Tolerable Waiting Time: How Long Are Web Users Willing to Wait?” In: *Behaviour & Information Technology - Behaviour & IT*. Jan. 1, 2003.

[134] Dennis Galletta, Raymond Henry, Scott McCoy, and Peter Polak. “Web Site Delays: How Tolerant Are Users?” In: *Journal of the Association for Information Systems* (Jan. 2004). URL: https://aisel.aisnet.org/jais/vol5/iss1/1/.

[135] Sam Van Damme, Javad Sameri, Susanna Schwarzmann, Qing Wei, Riccardo Trivisonno, Filip De Turck, and Maria Torres Vega. “Impact of Latency on QoE, Performance, and Collaboration in Interactive Multi-User Virtual Reality”. In: *Applied Sciences* (6 Jan. 2024). URL: https://www.mdpi.com/2076-3417/14/6/2290.

[136] Tom Beigbeder, Rory Coughlan, Corey Lusher, John Plunkett, Emmanuel Agu, and Mark Claypool. “The Effects of Loss and Latency on User Performance in Unreal Tournament 2003”. In: *Proceedings of 3rd ACM SIGCOMM Workshop on Network and System Support for Games*. Aug. 30, 2004. URL: https://dl.acm.org/doi/10.1145/1016540.1016556.

[137] Ivan Vaghi, Chris Greenhalgh, and Steve Benford. “Coping with Inconsistency Due to Network Delays in Collaborative Virtual Environments”. In: *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*. Dec. 20, 1999. URL: https://dl.acm.org/doi/10.1145/323663.323670.

[138] Cheryl Savery and T.C. Nicholas Graham. “It’s about Time: Confronting Latency in the Development of Groupware Systems”. In: *Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work*. Mar. 19, 2011. URL: https://dl.acm.org/doi/10.1145/1958824.1958851.

[139] Carl Gutwin, Steve Benford, Jeff Dyck, Mike Fraser, Ivan Vaghi, and Chris Greenhalgh. “Revealing Delay in Collaborative Environments”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Apr. 25, 2004. URL: https://dl.acm.org/doi/10.1145/985692.985756.

[140] Carl Gutwin, T.C. Graham, Christopher Wolfe, Nelson Wong, and Brian de Alwis. “Gone but Not Forgotten: Designing for Disconnection in Synchronous Groupware”. In: *Proceedings of the ACM Conference on Computer Supported Cooperative Work, CSCW*. Jan. 1, 2010.

[141] Yen-Ting Yeh, Nikhita Joshi, and Daniel Vogel. “The Effects of Update Interval and Reveal Method on Writer Comfort in Synchronized Shared-Editors”. In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*. May 11, 2024. URL: https://doi.org/10.1145/3613904.3642330.

[142] Shah Khalid, Aftab Alam, Muhammad Fayaz, Fakhrud Din, Sehat Ullah, and Shabir Ahmad. “Investigating the Effect of Network Latency on Users Performance in Collaborative Virtual Environments Using Navigation Aids”. In: *Future Generation Computer Systems* (Aug. 1, 2023). URL: https://www.sciencedirect.com/science/article/pii/S0167739X23000663.

[143] Vivek Dhakal, Anna Maria Feit, Per Ola Kristensson, and Antti Oulasvirta. “Observations on Typing from 136 Million Keystrokes”. In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. Apr. 21, 2018. URL: https://dl.acm.org/doi/10.1145/3173574.3174220.

[144] Peter Norvig. *English Letter Frequency Counts: Mayzner Revisited or ETAOIN SRHLDCU*. URL: https://norvig.com/mayzner.html (visited on 11/28/2023).

[145] Peter van Hardenberg and Martin Kleppmann. “PushPin: Towards Production-Quality Peer-to-Peer Collaboration”. In: *Proceedings of the 7th Workshop on Principles and Practice of Consistency for Distributed Data*. Apr. 27, 2020. URL: https://dl.acm.org/doi/10.1145/3380787.3393683.

[146] Matthieu Nicolas, Victorien Elvinger, Gérald Oster, Claudia-Lavinia Ignat, and François Charoy. “MUTE: A Peer-to-Peer Web-based Real-time Collaborative Editor”. In: *ECSCW 2017-15th European Conference on Computer-Supported Cooperative Work* (2017). URL: https://dl.eusset.eu/handle/20.500.12015/2950.

[147] Martin Kleppmann, Adam Wiggins, Peter van Hardenberg, and Mark McGranaghan. “Local-First Software: You Own Your Data, in Spite of the Cloud”. In: *Proceedings of the 2019 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*. Oct. 23, 2019. URL: https://dl.acm.org/doi/10.1145/3359591.3359737.

[148] Marc Shapiro and Nuno Preguiça. “Designing a Commutative Replicated Data Type”. Oct. 9, 2007. arXiv: 0710.1784 [cs]. URL: http://arxiv.org/abs/0710.1784.

[149] Gérald Oster, Pascal Urso, Pascal Molli, and Abdessamad Imine. “Data Consistency for P2P Collaborative Editing”. In: *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work - CSCW ’06*. 2006. URL: http://portal.acm.org/citation.cfm?doid=1180875.1180916.

[150] Mihai Letia, Nuno Preguiça, and Marc Shapiro. “CRDTs: Consistency without Concurrency Control”. July 6, 2009. arXiv: 0907.0929 [cs]. URL: http://arxiv.org/abs/0907.0929.

[151] Brice Nédelec, Pascal Molli, Achour Mostefaoui, and Emmanuel Desmontils. “LSEQ: An Adaptive Structure for Sequences in

Distributed Collaborative Editing". In: *Proceedings of the 2013 ACM Symposium on Document Engineering*. Sept. 10, 2013. URL: https://dl.acm.org/doi/10.1145/2494266.2494278.

[152] Stephane Weiss, Pascal Urso, and Pascal Molli. "Logoot: A Scalable Optimistic Replication Algorithm for Collaborative Editing on P2P Networks". In: *2009 29th IEEE International Conference on Distributed Computing Systems*. June 2009. URL: http://ieeexplore.ieee.org/document/5158450/.

[153] Martin Kleppmann and Alastair R. Beresford. "A Conflict-Free Replicated JSON Datatype". In: *IEEE Transactions on Parallel and Distributed Systems* (Oct. 1, 2017). URL: https://doi.org/10.1109/TPDS.2017.2697382.

[154] Mehdi Ahmed-Nacer, Claudia-Lavinia Ignat, Gérald Oster, Hyun-Gul Roh, and Pascal Urso. "Evaluating CRDTs for Real-Time Document Editing". In: *Proceedings of the 11th ACM Symposium on Document Engineering*. Sept. 19, 2011. URL: https://dl.acm.org/doi/10.1145/2034691.2034717.

[155] *Yjs Internals*. Apr. 7, 2022. URL: https://github.com/yjs/yjs/blob/8d809ebacbd648626c9909e03a19e10e02a7c190/INTERNALS.md (visited on 04/07/2022).

[156] Kevin Jahns. *Are CRDTs Suitable for Shared Editing?* Kevin's Blog. Aug. 10, 2020. URL: https://blog.kevinjahns.de/are-crdts-suitable-for-shared-editing/ (visited on 04/05/2022).

[157] Jelle van den Hooff, David Lazar, Matei Zaharia, and Nickolai Zeldovich. "Vuvuzela: Scalable Private Messaging Resistant to Traffic Analysis". In: *Proceedings of the 25th Symposium on Operating Systems Principles - SOSP '15*. 2015. URL: http://dl.acm.org/citation.cfm?doid=2815400.2815417.

[158] Nirvan Tyagi, Yossi Gilad, Derek Leung, Matei Zaharia, and Nickolai Zeldovich. “Stadium: A Distributed Metadata-Private Messaging System”. In: *Proceedings of the 26th Symposium on Operating Systems Principles*. Oct. 14, 2017. URL: https://dl.acm.org/doi/10.1145/3132747.3132783.

[159] David Lazar, Yossi Gilad, and Nickolai Zeldovich. “Karaoke: Distributed Private Messaging Immune to Passive Traffic Analysis”. In: *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*. 2018. URL: https://www.usenix.org/conference/osdi18/presentation/lazar.

[160] Albert Kwon, Henry Corrigan-Gibbs, Srinivas Devadas, and Bryan Ford. “Atom: Horizontally Scaling Strong Anonymity”. In: *Proceedings of the 26th Symposium on Operating Systems Principles*. Oct. 14, 2017. URL: https://dl.acm.org/doi/10.1145/3132747.3132755.

[161] Ludovic Barman, Moshe Kol, David Lazar, Yossi Gilad, and Nickolai Zeldovich. “Groove: Flexible {Metadata-Private} Messaging”. In: *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*. 2022. URL: https://www.usenix.org/conference/osdi22/presentation/barman.

[162] David Lazar, Yossi Gilad, and Nickolai Zeldovich. “Yodel: Strong Metadata Security for Voice Calls”. In: *Proceedings of the 27th ACM Symposium on Operating Systems Principles*. Oct. 27, 2019. URL: https://dl.acm.org/doi/10.1145/3341301.3359648.

[163] Raymond Cheng, William Scott, Elisaweta Masserova, Irene Zhang, Vipul Goyal, Thomas Anderson, Arvind Krishnamurthy, and Bryan Parno. “Talek: Private Group Messaging with Hidden Access Patterns”. In: *Annual Computer Security Applications Conference*. Dec. 8, 2020. URL: https://dl.acm.org/doi/10.1145/3427228.3427231.

[164] The Tor Project. *Tor Performance*. 2024. URL: https://metrics.torproject.org/onionperf-latencies.html?start=2023-09-11&end=2023-10-09&server=public (visited on 02/02/2024).

[165] Ania M. Piotrowska. "Studying the Anonymity Trilemma with a Discrete-event Mix Network Simulator". In: *Proceedings of the 20th Workshop on Workshop on Privacy in the Electronic Society*. Nov. 15, 2021. URL: https://dl.acm.org/doi/10.1145/3463676.3485614.

[166] Iness Ben Guirat, Debajyoti Das, and Claudia Diaz. "Blending Different Latency Traffic With Beta Mixing". In: *Proceedings on Privacy Enhancing Technologies* (Apr. 2024). URL: https://petsymposium.org/popets/2024/popets-2024-0059.php.

[167] Iness Ben Guirat, Devashish Gosain, and Claudia Diaz. "MiXiM: Mixnet Design Decisions and Empirical Evaluation". In: *Proceedings of the 20th Workshop on Workshop on Privacy in the Electronic Society*. Nov. 15, 2021. URL: https://dl.acm.org/doi/10.1145/3463676.3485613.

[168] Lennart Oldenburg, Marc Juarez, Enrique Argones Rúa, and Claudia Diaz. "MixMatch: Flow Matching for Mixnet Traffic". In: *Proceedings on Privacy Enhancing Technologies* (Apr. 2024). URL: https://petsymposium.org/popets/2024/popets-2024-0050.php.

[169] Sarah Abdelwahab Gaballah, Lamya Abdullah, Minh Tung Tran, Ephraim Zimmer, and Max Mühlhäuser. "On the Effectiveness of Intersection Attacks in Anonymous Microblogging". In: *Secure IT Systems: 27th Nordic Conference, NordSec 2022,* Jan. 1, 2023. URL: https://doi.org/10.1007/978-3-031-22295-5_1.

[170] Daniel Hugenroth and Alastair R Beresford. "Powering Privacy: On the Energy Demand and Feasibility of Anonymity Networks on Smartphones". In: *32nd USENIX Security Symposium (USENIX Security 23)*. 2023.

[171] Killian Davitt. *Mixnet User Study Software*. GitHub. URL: https://github.com/KillianDavitt/mixnet-user-study (visited on 12/22/2023).

[172] Elissa M. Redmiles, Yasemin Acar, Sascha Fahl, and Michelle L. Mazurek. *A Summary of Survey Methodology Best Practices for Security and Privacy Researchers*. University of Maryland, May 3, 2017. URL: http://hdl.handle.net/1903/19227.

[173] Paul A. Games and John F. Howell. "Pairwise Multiple Comparison Procedures with Unequal Ns and/or Variances: A Monte Carlo Study". In: *Journal of Educational Statistics* (June 1, 1976). URL: https://doi.org/10.3102/10769986001002113.

[174] Mital C Shingala. "Comparison of Post Hoc Tests for Unequal Variance". In: *International Journal of New Technologies in Science and Engineering* (2015).

[175] Christoph Kerschbaumer, Julian Gaibler, Arthur Edelstein, and Thyla van der Merwey. "HTTPS-Only: Upgrading All Connections to Https in Web Browsers". In: *Proceedings 2021 Workshop on Measurements, Attacks, and Defenses for the Web*. 2021. URL: https://www.ndss-symposium.org/wp-content/uploads/madweb2021_23010_paper.pdf.

[176] Emily Stark and Carlos Joan Rafael Ibarra Lopez. *No More Mixed Messages About HTTPS*. Chromium Blog. Oct. 3, 2019. URL: https://blog.chromium.org/2019/10/no-more-mixed-messages-about-https.html (visited on 08/09/2022).

[177] Emily Schechter. *A Milestone for Chrome Security: Marking HTTP as Not Secure*. Google. July 24, 2018. URL: https://blog.google/products/chrome/milestone-chrome-security-marking-http-not-secure/ (visited on 08/15/2022).

[178] Scott Helme. *Top 1 Million Analysis - November 2021*. Top 1 Million Analysis - November 2021. Dec. 9, 2021. URL: https://scotthelme.co.uk/top-1-million-analysis-november-2021/ (visited on 11/08/2023).

[179] *Brave: Secure, Fast, & Private Web Browser with Adblocker*. Brave. URL: https://brave.com/ (visited on 10/15/2023).

[180] *Strict HTTPS Upgrade Mode in Brave Browser*. Brave Help Center. May 5, 2023. URL: https://support.brave.com/hc/en-us/articles/15513090104717-Strict-HTTPS-Upgrade-Mode-in-Brave-Browser (visited on 08/22/2023).

[181] *Change HTTPS Only Mode Error Copy to Give the Sense of "Protection"*. 2020. URL: https://bugzilla.mozilla.org/show_bug.cgi?id=1644146 (visited on 11/21/2023).

[182] Jacki Schirmer. "Ethical Issues in the Use of Multiple Survey Reminders". In: *Journal of Academic Ethics* (June 2009). URL: http://link.springer.com/10.1007/s10805-009-9072-5.

[183] Adrienne Porter Felt, Alex Ainslie, Robert W. Reeder, Sunny Consolvo, Somas Thyagaraja, Alan Bettes, Helen Harris, and Jeff Grimes. "Improving SSL Warnings: Comprehension and Adherence". In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. Apr. 18, 2015. URL: https://dl.acm.org/doi/10.1145/2702123.2702442.

[184] Mozilla. *Privilege Escalation via DOM Property Overrides*. Mozilla. 2005. URL: https://www.mozilla.org/en-US/security/advisories/mfsa2005-41/ (visited on 01/12/2024).

[185] Gareth Heyes. *Escaping JavaScript Sandboxes with Parsing Issues*. PortSwigger Research. July 10, 2020. URL:

https://portswigger.net/research/escaping-javascript-sandboxes-with-parsing-issues (visited on 01/12/2024).

[186] Marco Bartoli Candreva Giulio. *Escaping the Sandbox: A Bug That Speaks for Itself*. Microsoft Browser Vulnerability Research. Nov. 14, 2023. URL: https://microsoftedge.github.io/edgevr/posts/Escaping-the-sandbox-A-bug-that-speaks-for-itself/ (visited on 01/12/2024).

[187] The Tor Project. *NoScript | Tor Project*. URL: https://support.torproject.org/glossary/noscript/ (visited on 10/15/2023).

[188] Tor Project. *JavaScript Enabled by Default in Tor Browser*. Tor Project. URL: https://support.torproject.org/#tbb_tbb-34 (visited on 10/15/2023).

[189] Iacovos Kirlappos and M. Angela Sasse. "Security Education against Phishing: A Modest Proposal for a Major Rethink". In: *IEEE Security & Privacy Magazine* (Mar. 2012). URL: http://ieeexplore.ieee.org/document/6109230/.

[190] Lujo Bauer, Cristian Bravo-Lillo, Lorrie Cranor, and Elli Fragkaki. *Warning Design Guidelines*. Carnegie Mellon University, 2013.

[191] Adrienne Porter Felt, Robert W. Reeder, Hazim Almuhimedi, and Sunny Consolvo. "Experimenting at Scale with Google Chrome's SSL Warning". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Apr. 26, 2014. URL: https://doi.org/10.1145/2556288.2557292.

[192] Benjamin Greschbach, Tobias Pulls, Laura M. Roberts, Philipp Winter, and Nick Feamster. "The Effect of DNS on Tor's Anonymity". In: *Proceedings 2017 Network and Distributed System Security Symposium*. 2017. URL: https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/e-effect-dns-tors-anonymity/.

[193] Joe DeBlasio. *Towards HTTPS by Default*. Chromium Blog. URL: https://blog.chromium.org/2023/08/towards-https-by-default.html (visited on 08/22/2023).

[194] Joe DeBlasio. *Protecting Users from Insecure Downloads in Google Chrome*. Chromium Blog. 2020. URL: https://blog.chromium.org/2020/02/protecting-users-from-insecure.html (visited on 10/17/2023).

[195] Luka Jelovcan, Simon L R Vrhovec, and Anze Mihelic. "A Literature Survey of Security Indicators in Web Browsers". In: *Elektrotehniki vestnik* (2020).

[196] Devdatta Akhawe and Adrienne Porter Felt. "Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness". In: *22nd USENIX Security Symposium (USENIX Security 13)*. 2013. URL: https://www.usenix.org/conference/usenixsecurity13/technical-sessions/presentation/akhawe.

[197] Cristian Bravo-Lillo, Saranga Komanduri, Lorrie Faith Cranor, Robert W. Reeder, Manya Sleeper, Julie Downs, and Stuart Schechter. "Your Attention Please: Designing Security-Decision UIs to Make Genuine Risks Harder to Ignore". In: *Proceedings of the Ninth Symposium on Usable Privacy and Security - SOUPS '13*. 2013. URL: http://dl.acm.org/citation.cfm?doid=2501604.2501610.

[198] Rainer Böhme and Stefan Köpsell. "Trained to Accept? A Field Experiment on Consent Dialogs". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Apr. 10, 2010. URL: https://dl.acm.org/doi/10.1145/1753326.1753689.

[199] Cristian Bravo-Lillo, Lorrie Faith Cranor, Julie Downs, and Saranga Komanduri. "Bridging the Gap in Computer Security Warnings: A Mental Model Approach". In: *IEEE Security & Privacy* (Mar. 2011).

[200] Robert W. Reeder, Adrienne Porter Felt, Sunny Consolvo, Nathan Malkin, Christopher Thompson, and Serge Egelman. "An Experience Sampling Study of User Reactions to Browser Warnings in the Field". In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. Apr. 21, 2018. URL: https://doi.org/10.1145/3173574.3174086.

[201] Franziska Herbert, Steffen Becker, Leonie Schaewitz, Jonas Hielscher, Marvin Kowalewski, Angela Sasse, Yasemin Acar, and Markus Dürmuth. "A World Full of Privacy and Security (Mis)Conceptions? Findings of a Representative Survey in 12 Countries". In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. Apr. 19, 2023. URL: https://dl.acm.org/doi/10.1145/3544548.3581410.

[202] Serge Egelman and Stuart Schechter. "The Importance of Being Earnest [In Security Warnings]". In: *Financial Cryptography and Data Security*. Red. by David Hutchison et al. 2013. URL: http://link.springer.com/10.1007/978-3-642-39884-1_5.

[203] Hazim Almuhimedi, Adrienne Porter Felt, Robert W. Reeder, and Sunny Consolvo. "Your Reputation Precedes You: History, Reputation, and the Chrome Malware Warning". In: *10th Symposium On Usable Privacy and Security (SOUPS 2014)*. 2014. URL: https://www.usenix.org/conference/soups2014/proceedings/presentation/almuhimedi.

[204] Ruogu Kang, Laura Dabbish, Nathaniel Fruchter, and Sara Kiesler. "My Data Just Goes Everywhere:User Mental Models of the Internet and Implications for Privacy and Security". In: *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*. 2015. URL: https://www.usenix.org/conference/soups2015/proceedings/presentation/kang.

[205] Batya Friedman, David Hurley, Daniel C. Howe, Edward Felten, and Helen Nissenbaum. "Users' Conceptions of Web Security: A Comparative Study". In: *CHI '02 Extended Abstracts on Human Factors in Computing*

Systems. Apr. 20, 2002. URL: https://dl.acm.org/doi/10.1145/506443.506577.

[206] Holly E. Hancock, C. Travis Bowles, Wendy A. Rogers, and Arthur D. Fisk. "Comprehension and Retention of Warning Information". In: *Handbook of Warnings*. 2006.

[207] Kenneth R. Laughery and Julie A. Stanush. "Effects of Warning Explicitness on Product Perceptions". In: *Proceedings of the Human Factors Society Annual Meeting* (Oct. 1, 1989). URL: https://doi.org/10.1518/107118189786759732.

[208] Joshua Sunshine, Serge Egelman, Hazim Almuhimedi, Neha Atri, and Lorrie Faith Cranor. "Crying Wolf: An Empirical Study of SSL Warning Eectiveness". In: *18th USENIX Security Symposium*. Aug. 2009.

[209] Jackson Stokes, Tal August, Robert A Marver, Alexei Czeskis, Franziska Roesner, Tadayoshi Kohno, and Katharina Reinecke. "How Language Formality in Security and Privacy Interfaces Impacts Intended Compliance". In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. Apr. 19, 2023. URL: https://dl.acm.org/doi/10.1145/3544548.3581275.

[210] Joel Weinberger and Adrienne Porter Felt. "A Week to Remember: The Impact of Browser Warning Storage Policies". In: *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*. 2016. URL: https://www.usenix.org/conference/soups2016/technical-sessions/presentation/weinberger.

[211] *Entering Public Beta - Let's Encrypt*. 2015. URL: https://letsencrypt.org/2015/12/03/entering-public-beta.html (visited on 12/01/2023).

[212] Sudheesh Singanamalla, Esther Han Beol Jang, Richard Anderson, Tadayoshi Kohno, and Kurtis Heimerl. "Accept the Risk and Continue: Measuring the Long Tail of Government Https Adoption". In: *Proceedings*

of the ACM Internet Measurement Conference. Oct. 27, 2020. URL: https://doi.org/10.1145/3419394.3423645.

[213] Katharina Krombholz, Wilfried Mayer, Martin Schmiedecker, and Edgar Weippl. “"I Have No Idea What I’m Doing" - On the Usability of Deploying HTTPS”. In: *26th USENIX Security Symposium (USENIX Security 17)*. 2017. URL: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/krombholz.

[214] Andreas Sotirakopoulos, Kirstie Hawkey, and Konstantin Beznosov. “On the Challenges in Usable Security Lab Studies: Lessons Learned from Replicating a Study on SSL Warnings”. In: *Proceedings of the Seventh Symposium on Usable Privacy and Security*. July 20, 2011. URL: https://dl.acm.org/doi/10.1145/2078827.2078831.

[215] Kat Krol, Jonathan M. Spring, Simon Parkin, and M. Angela Sasse. “Towards Robust Experimental Design for User Studies in Security and Privacy”. In: 2016. URL: https://www.usenix.org/conference/laser2016/program/presentation/krol.

[216] Paul Syverson and Matthew Traudt. “HSTS Supports Targeted Surveillance”. In: *8th USENIX Workshop on Free and Open Communications on the Internet (FOCI 18)}*. 2018. URL: https://www.usenix.org/conference/foci18/presentation/syverson.

[217] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. “RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response”. In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. Nov. 3, 2014. URL: https://dl.acm.org/doi/10.1145/2660267.2660348.

[218] Johann Hofmann and Tim Huang. *Introducing State Partitioning*. Mozilla Hacks the Web developer blog. Feb. 23, 21. URL: https:

//hacks.mozilla.org/2021/02/introducing-state-partitioning (visited on 02/14/2023).

[219] *Protecting Against HSTS Abuse*. WebKit. Mar. 16, 2018. URL: https://webkit.org/blog/8146/protecting-against-hsts-abuse/ (visited on 02/24/2023).

[220] Yan Zhu. *Weird New Tricks for Browser Fingerprinting*. 2015. URL: https://zyan.scripts.mit.edu/presentations/toorcon2015.pdf (visited on 11/24/2021).

[221] Yan Zhu. *Sniffly2*. Oct. 8, 2023. URL: https://github.com/diracdeltas/sniffly.

[222] *Browser History Sniffing Attack Using HSTS + CSP*. URL: https://bugs.chromium.org/p/chromium/issues/detail?id=544765 (visited on 11/03/2023).

[223] Alexander Færøy. *Tor Directory Protocol, Version 3*. 2023. URL: https://gitweb.torproject.org/torspec.git/tree/dir-spec.txt (visited on 05/31/2023).

[224] Tor Project. *Bad Relays*. 2023. URL: https://community.torproject.org/relay/community-resources/bad-relays/ (visited on 06/12/2023).

[225] The Tor Project. *Tor Browser HSTS Preload List*. Oct. 6, 2022. URL: https://gitlab.torproject.org/tpo/applications/tor-browser/-/blob/tor-browser-102.4.0esr-12.0-2/security/manager/ssl/nsSTSPreloadList.inc (visited on 06/13/2023).

[226] The Tor Project. *Releases: Tor Browser 11.5*. 2022. URL: https://www.torproject.org/releases/tor-browser-11-5/.

[227] Cynthia Dwork. "Differential Privacy". In: *Automata, Languages and Programming*. 2006.

[228] S. L. Warner. “Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias”. In: *Journal of the American Statistical Association* (Mar. 1965). pmid: 12261830.

[229] Chromium Design Documents. *Rappor (Randomized Aggregatable Privacy Preserving Ordinal Responses)*. Sept. 3, 2018. URL: https://web.archive.org/web/20180309120825/https://www.chromium.org/developers/design-documents/rappor (visited on 05/30/2023).

[230] *HSTS Study*. URL: https://hstsadoption.github.io/#anomalies (visited on 12/02/2021).

[231] Lucas Garron, Andrew Bortz, and Dan Boneh. *The State of HSTS Deployment: A Survey and Common Pitfalls*. 2013. Pre-published.

[232] *Usage Statistics of HTTP Strict Transport Security for Websites, July 2022*. URL: https://w3techs.com/technologies/details/ce-hsts (visited on 07/25/2022).

[233] Sergio de los Santos, Carmen Torrano, Yaiza Rubio, and Félix Brezo. “Implementation State of HSTS and HPKP in Both Browsers and Servers”. In: *Cryptology and Network Security*. 2016.

[234] Sergio De los Santos and José Torres. “Analysing HSTS and HPKP Implementation in Both Browsers and Servers”. In: *IET Information Security* (July 2018). URL: https://onlinelibrary.wiley.com/doi/10.1049/iet-ifs.2017.0030.

[235] The ZMap Team. *ZGrab 2.0*. July 25, 2022. URL: https://github.com/zmap/zgrab2 (visited on 07/25/2022).

[236] Michael Kranch and Joseph Bonneau. “Upgrading HTTPS in Mid-Air: An Empirical Study of Strict Transport Security and Key Pinning”. In: *Proceedings 2015 Network and Distributed System Security Symposium*. 2015. URL: https://www.ndss-symposium.org/ndss2015/ndss-

2015-programme/upgrading-http-mid-air-empirical-study-strict-transport-security-and-key-pinning/.

[237] *Tor Metrics*. 2022. URL: https://metrics.torproject.org/userstats-relay-table.html (visited on 05/18/2022).

[238] *We Retired Alexa.Com on May 1, 2022*. Alexa Support. 2022. URL: https://support.alexa.com/hc/en-us/articles/4410503838999-We-retired-Alexa-com-on-May-1-2022 (visited on 07/25/2022).

[239] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. "Privacy Pass: Bypassing Internet Challenges Anonymously". In: *Proceedings on Privacy Enhancing Technologies* (June 1, 2018). URL: https://petsymposium.org/popets/2018/popets-2018-0026.php.

[240] Tommy Pauly, Steven Valdez, and Christopher A. Wood. *The Privacy Pass HTTP Authentication Scheme*. Internet Draft. Internet Engineering Task Force, May 8, 2023. URL: https://datatracker.ietf.org/doc/draft-ietf-privacypass-auth-scheme.

[241] Intuition Machines. *hCaptcha - Stop More Bots. Start Protecting Privacy.* 2022. URL: https://www.hcaptcha.com/ (visited on 01/12/2022).

[242] Ari Juels and John Brainard. "Client Puzzles: A Cryptographic Countermeasure against Connection Depletion Attacks". In: *Proceedings of NDSS '99*. 1999.

[243] Mayank Raikwar and Danilo Gligoroski. "Non-Interactive VDF Client Puzzle for DoS Mitigation". In: *European Interdisciplinary Cybersecurity Conference*. Nov. 10, 2021. URL: https://dl.acm.org/doi/10.1145/3487405.3487406.

[244] Google. *Rappor Implementation*. GitHub, 2017. URL: https://github.com/google/rappor.

[245] Killian Davitt and Dan Ristea. *CoStricTor*. Aug. 25, 2023. URL: https://github.com/KillianDavitt/CoStricTor.

[246] Aniket Mahanti, Niklas Carlsson, Anirban Mahanti, Martin Arlitt, and Carey Williamson. "A Tale of the Tails: Power-laws in Internet Measurements". In: *IEEE Network* (Jan. 2013). URL: http://ieeexplore.ieee.org/document/6423193/.

[247] Serge A. Krashakov, Anton B. Teslyuk, and Lev N. Shchur. "On the Universality of Rank Distributions of Website Popularity". In: *Computer Networks* (Aug. 10, 2006). URL: https://www.sciencedirect.com/science/article/pii/S1389128605002513.

[248] Jakob Nielsen. *Traffic Log Patterns*. Nielsen Norman Group. URL: https://www.nngroup.com/articles/traffic-log-patterns/ (visited on 05/26/2022).

[249] *Tor Traffic Metrics*. 2023. URL: https://metrics.torproject.org/bandwidth-flags.html?start=2022-04-26&end=2023-06-12 (visited on 06/12/2023).

[250] 2captcha.com. *Captcha Prices*. Aug. 18, 2023. URL: https://web.archive.org/web/20230818150417/https://2captcha.com/pricing (visited on 08/18/2023).

[251] Giulia Fanti, Vasyl Pihur, and Úlfar Erlingsson. "Building a RAPPOR with the Unknown: Privacy-Preserving Learning of Associations and Data Dictionaries". In: *Proceedings on Privacy Enhancing Technologies* (July 1, 2016). URL: https://petsymposium.org/popets/2016/popets-2016-0015.php.

[252] Henry Corrigan-Gibbs and Dan Boneh. "Prio: Private, Robust, and Scalable Computation of Aggregate Statistics". In: *8th USENIX Workshop on Free and Open Communications on the Internet (FOCI 18)*. 2017. URL:

https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/corrigan-gibbs.

[253] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. "Collecting Telemetry Data Privately". In: *Advances in Neural Information Processing Systems*. 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/hash/253614bbac999b38b5b60cae531c4969-Abstract.html.

[254] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. "Prochlo: Strong Privacy for Analytics in the Crowd". In: *Proceedings of the 26th Symposium on Operating Systems Principles*. Oct. 14, 2017. URL: https://dl.acm.org/doi/10.1145/3132747.3132769.

[255] Alex Davidson, Peter Snyder, E. B. Quirk, Joseph Genereux, Benjamin Livshits, and Hamed Haddadi. "STAR: Secret Sharing for Private Threshold Aggregation Reporting". In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. Nov. 7, 2022. arXiv: 2109.10074 [cs]. URL: http://arxiv.org/abs/2109.10074.

[256] Tariq Elahi, George Danezis, and Ian Goldberg. "PrivEx: Private Collection of Traffic Statistics for Anonymous Communication Networks". In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. Nov. 3, 2014. URL: https://dl.acm.org/doi/10.1145/2660267.2660280.

[257] Rasmus Dahlberg, Tobias Pulls, Tom Ritter, and Paul Syverson. "Privacy-Preserving & Incrementally-Deployable Support for Certificate Transparency in Tor". In: *Proceedings on Privacy Enhancing Technologies* (Apr. 1, 2021). URL: https://www.sciendo.com/article/10.2478/popets-2021-0024.

Appendix A

High-latency Anonymity System Delay User Study Questions

Section 1: Information & Consent

- **Title of Study:** Evaluating the effect of latency on collaboration
- **Department:** Department of Computer Science, UCL
- **Researcher:** Killian Davitt, [redacted]
- **Principal Researcher:** Prof. Steven Murdoch, [redacted]
- **UCL Data Protection Officer:** data-protection@ucl.ac.uk

Invitation to take part You are being invited to take part in this research project. Before you continue it is important for you to understand why the research us being done and what participation will involve. Please take time to read the following information carefully and discuss it with others if you wish. Please contact us via prolific if there is anything that is not clear or if you would like more information. Take time to decide whether or not you wish to take part. Thank you for reading.

What is the projects purpose? This study is attempting to evaluate the effect different levels of delay has on collaboration online. Anonymous communication online usually involves adding delay to communications, therefore, we wish to find out how much delay can be added to collaboration before communication becomes difficult.

Why have I been chosen? You have been recruited for this survey through prolific. We are attempting to recruit a balanced demographic of users by age and sex, and this may have influenced why you have been recruited.

Do I have to take part? The study is completely voluntary and you should feel free to discontinue the survey at any time without consequence. If you wish to stop the survey at any time, please simply close the website.

What will happen to me if I take part? The study should take approximately 6 minutes to complete. You will be answering a series of simple questions in collaboration with another simulated "fake" user. This computer script will be assisting you in answering some of the questions you are asked. The delay that is added may make this more difficult as you and the simulated user may attempt to answer the same question at the same time.

When you have answered each set of questions you will be asked to grade your experience depending on how frustrating it was, and how well able you were to collaborate with the simulated user. The data we collect from this will then help us to understand how much delay can be added before the experience worsens for users of a collaborative system like this.

What are the possible disadvantages and risks of taking part? As a participant you will face no additional risks, other than the risk typically encountered by using your computer.

What are the possible benefits of taking part? Payment for this survey will be £0.75

What if something goes wrong? If you have any complaints or questions about the survey, please contact us through prolific in the first instance, or via the email addresses above. If you do not feel your complain has been properly resolve, you can contact the chair of the UCL ethics committee at ethics@ucl.ac.uk

Will my taking part in this project be kept confidential? All data collected in this survey will be subject to GDPR protections and will only be released in a form which cannot identify you. None of the data collected can identify you.

What will happen to the results of the research project? The results of this study will be used in our academic work and may be published at a later date. More information about the results will be found here when it is available. `https://killiandavitt.me/collaboration-user-study`

Data protection notice

The data controller for this project will be University College London (UCL). The UCL Data Protection Officer provides oversight of UCL activities involving the processing of personal data, and can be contacted at data-protection@ucl.ac.uk

This local privacy notice sets out the information that applies to this particular study. Further information on how UCL uses participant information can be found in our general privacy notice: `https://www.ucl.ac.uk/legal-services/privacy/ucl-general-research-participant-privacy-notice`

The information that is required to be provided to participants under data protection legislation (GDPR and DPA 2018) is provided across both the local and general privacy notices.

As a result of your participation in this survey, we will receive your anonymous demographic data which you have previously provided to Prolific. We will also ask you for your approximate education level. The time taken for you to complete each question will also be recorded. All data collected in this survey will be subject to GDPR protections and will only be released in a form which cannot identify you.

The lawful basis that will be used to process your personal data will be performance of a task in the public interest. We do not seek any personal data in this survey, and you should not enter personal data into any of the textboxes.

The lawful basis that would be used to process any special category data will be scientific or historical research purposes. As this survey does not seek special category data, this lawful basis is only relevant to data which will be deleted by the researchers. I.e. if special category data is entered by participants, it will be deleted before any further processing.

Please do not enter personal data into the text boxes, this ensures we will only receive data from you that is anonymous. If you are concerned about how your data is being processed, or if you would like to contact us about your rights, please

contact UCL in the first instance at data-protection@ucl.ac.uk.

If you wish to revoke your consent for the processing of your data, you can revoke consent through the Prolific platform, or contact us at the email below. If you revoke your consent after completing the survey, you will still be paid. If you have any complaints or questions about the survey, please contact us through prolific in the first instance, or via the email addresses below. If you do not feel your complain has been properly resolve, you can contact the chair of the UCL ethics committee at ethics@ucl.ac.uk

Who is organising and funding the research? This study is being conducted by the UCL Information Security Group which is part of the Department of Computer Science. The study is funded by the Royal Society.

Other information Please also note that before moving on to the study we will first check that you understand the task clearly and may at that point terminate the survey at our discretion. You will not receive payment in this case.

This study has been given ethical approval by the Head of Department of UCL Computer Science under id: UCL/CSREC/R/28

We thank you for considering participating in our study. If you have any other questions please contact us through the prolific website contact button.

Before continuing please feel free to save a copy of this information by downloading it here: `https://killiandavitt.me/files/study_information.pdf`

- ☐ I understand that my participation is voluntary and that I am free to withdraw at any time without giving a reason

- ☐ I understand that my data gathered in this study will be stored anonymously and securely. It will not be possible to identify me in any publications.

- ☐ I understand that if I decide to withdraw, any personal data I have provided up to that point will be deleted unless I agree otherwise

☐ I am aware of who I should contact if I wish to lodge a complaint

☐ I voluntarily agree to take part in this study.

☐ I confirm that I am over 18 years of age

Section 2

Please enter your current highest level of formal education

- ☐ None, or incomplete secondary education
- ☐ Secondary education (A levels, GCSE, further education, etc)
- ☐ University degree (BA, BSc, etc)
- ☐ Graduate degree (Masters, PhD, etc)

Section 3

This survey will involve answering a series of general knowledge questions. You will be faced with 5 different question sets, each containing 10 questions. Each time you will see all ten questions at once and you can answer them in any order you wish.

A simulated fake second user will be assisting you in answering these questions. This scripted user will randomly attempt to answer some of the questions for you. It will try to not answer questions you have already answered, or questions you are currently filling out. Similarly, you should not try to answer questions the simulated user is currently answering. Each question set however, will have a different level of added delay which will affect communication between you and the simulated user. Longer delays may mean you accidentally collide with each other, for example you may try and answer the same question at the same time which can cause difficulty.

The goal of this study is to evaluate how well you can work with the simulated user at different levels of delay. It does not matter how long you take to finish, and you should not worry about getting the correct answer to every question. If you are unsure of an answer, you can submit your best guess.

After each question set you will be given a very short questionnaire to ask you about your experience answering the questions.

Please do not enter any personal information into any of the text boxes. This will cause your submission to be deleted.

Please answer the following questions to ensure you have understood the survey:

Who will be assisting you in answering the questions you are faced with?

- ☐ Another researcher from our team
- ☐ A fake simulated user
- ☐ Another survey participant

What is this survey attempting to measure?

- ☐ The ability of participants to answer general knowledge questions
- ☐ Whether humans can beat AI at answering questions
- ☐ How collaboration is affected by added communication delays

Section 4

(Participants are directed to this section if they answer incorrectly the previous question)

Please read these instructions carefully and attempt to answer the question again

This survey will involve answering a series of general knowledge questions. You will be faced with 5 different question sets, each containing 10 questions. Each time you will see all ten questions at once and you can answer them in any order you wish.

A simulated fake second user will be assisting you in answering these questions. This scripted user will randomly attempt to answer some of the questions for you. It will try to not answer questions you have already answered, or questions you are currently filling out. Similarly, you should not try to answer questions the simulated user is currently answering. Each question set however, will have a different level of added delay which will affect communication between you and the simulated user. Longer delays may mean you accidentally collide with each other, for example you may try and answer the same question at the same time which can cause difficulty.

The goal of this study is to evaluate how well you can work with the simulated user at different levels of delay. It does not matter how long you take to finish, and you should not worry about getting the correct answer to every question. If you are unsure of an answer, you can submit your best guess.

After each question set you will be given a very short questionnaire to ask you about your experience answering the questions.

Please do not enter any personal information into any of the text boxes. This will cause your submission to be deleted.

Please answer the following questions to ensure you have understood the survey:

Who will be assisting you in answering the questions you are faced with?

☐ Another researcher from our team

☐ A fake simulated user

☐ Another survey participant

What is this survey attempting to measure?

☐ The ability of participants to answer general knowledge questions

☐ Whether humans can beat AI at answering questions

☐ How collaboration is affected by added communication delays

Section 5

(Participants are directed to this section if they answer incorrectly the previous question)

Unfortunately, we have not managed to convey the background of our survey to you successfully and we cannot continue the study. Please close the survey and return to Prolific and select 'Stop Without Completing'. Unfortunately we will not provide payment in this case.

– Survey ends here –

Section 6 - Main Questions

(**Note:** The following questions will be presented in a random order to the participant, questions also appear gradually to participants)

Please fill out these questions

This is task 1 of 6. More questions will slowly appear as you answer the questions.

Please provide the capital city of the following countries

What is the capital of China?

What colour is snow?

What language is spoken in Germany?

What language is spoken in England?

What is twenty plus twenty?

What is the capital of France?

What language is spoken in Japan?

Who is the UK Prime Minister?

What is the capital of Japan?

Which city in Germany was divided by a wall until 1989?

Which item do some people wear on their eyes to improve their eyesight

Which commonly eaten sweet food is typicaly brown and comes in bar form?

What language is most spoken in the USA?

Which language is spoken in Spain?

Please answer some questions on the experience you just had with the second user

I found working with the second user frustrating:

Not frustrating at all	a tiny bit frustrating	Slightly frustrating	very frustrating	Highly frustrating, not usable

What effect did the 2nd user have on how long it took to complete the task:

Much slower	Slightly slower	No change	slightly quicker	Much more quickly

Did you have to adapt your actions because the 2nd user was unpredictable? If yes, please describe how below.

- ○ Yes
- ○ No

Please leave a few words to review your experience of working with the second user:

Next

Please fill out these questions

This is task 2 of 6. More questions will slowly appear as you answer the questions.

fourteen, fifteen, ___, seventeen	
What is the opposite of day?	
Who was the leader of Germany during World War 2?	
In which country would you find the pyramids?	
Which alcoholic beverage is made from grapes?	
What is twenty minus five?	
What is the biggest country in the UK? England	
What colour is the sky?	
What is the capital of Italy?	
What is seven plus seven?	
Who was the first man on the moon?	
Which very popular messaging app has a green logo?	
Which superhero has the appearance of a bat?	
What item worn on the hand/arm tells the time?	

Please answer some questions on the experience you just had with the second user

I found working with the second user frustrating:

Not frustrating at all	a tiny bit frustrating	Slightly frustrating	very frustrating	Highly frustrating, not usable

What effect did the 2nd user have on how long it took to complete the task:

Much slower	Slightly slower	No change	slightly quicker	Much more quickly

Did you have to adapt your actions because the 2nd user was unpredictable? If yes, please describe how below.

- ○ Yes
- ○ No

Please leave a few words to review your experience of working with the second user:

Next

Please fill out these questions

This is task 3 of 6. More questions will slowly appear as you answer the questions.

Please write the word ‘Washington’ backwards.	
Please write the word ‘Philadelphia’ backwards.	
The United States of __	
Which country is famous for the ‘Great Wall’?	
What colour is a banana?	
What continent is South Africa in?	
What is a three sided shape called?	
Who is the president of the United States?	
Please write the word ‘elephant’ backwards.	
Which orange vegtable is commonly eaten in the UK?	
Which Ocean is on the west coast of the USA?	
What do caterpillars turn into?	
What phenomenon makes things fall to the ground?	
What white fluffy thing appears in the sky?	

Please answer some questions on the experience you just had with the second user

I found working with the second user frustrating:

Not frustrating at all	a tiny bit frustrating	Slightly frustrating	very frustrating	Highly frustrating, not usable

What effect did the 2nd user have on how long it took to complete the task:

Much slower	Slightly slower	No change	slightly quicker	Much more quickly

Did you have to adapt your actions because the 2nd user was unpredictable? If yes, please describe how below.

- ○ Yes
- ○ No

Please leave a few words to review your experience of working with the second user:

Next

Please fill out these questions

This is task 4 of 6. More questions will slowly appear as you answer the questions.

one, two, __, four, five.	
What is the capital of the UK?	
Please write the word ‘Poland’ backwards.	
What colour is grass?	
What is eight minus four?	
What shape has four sides?	
Which city is the headquarters of most European Union organisations?	
What is the name of the planet we live on?	
Which country is north of the USA?	
Which month comes after July?	
Which liquid falls from the sky as rain?	
Which ocean separates Europe and America? Atl_____	
Which large ship famously sank in 1912 after hitting an iceberg? Ti_____	
What language is spoken in Portugal?	

Please answer some questions on the experience you just had with the second user

I found working with the second user frustrating:

Not frustrating at all	a tiny bit frustrating	Slightly frustrating	very frustrating	Highly frustrating, not usable

What effect did the 2nd user have on how long it took to complete the task:

Much slower	Slightly slower	No change	slightly quicker	Much more quickly

Did you have to adapt your actions because the 2nd user was unpredictable? If yes, please describe how below.

- ○ Yes
- ○ No

Please leave a few words to review your experience of working with the second user:

Next

Please fill out these questions

This is task 5 of 6. More questions will slowly appear as you answer the questions.

What is the first name of the King of the UK?

Which Marvel superhero has the powers of a Spider?

Which country is referred to as ‘down under’: Au_____

Which British author wrote Hamlet? Wi_____ Sha_____

Which English speaking country is nearby Australia? N_w Z_____

Which white coloured drink is obtained from Cows?

In which Palace does the British King live in London?

Which famous clock is attached to the houses of parliament in London?

There are two main types of smartphone: Iphone and A____d

On which social network are ‘Tweets’ found?

Which social network was founded by Mark Zuckerburg?

Which social network is commonly referred to as ‘Insta’?

Which video streaming service has a red ‘N’ as its logo?

Which large online retailer shares it’s name with a large river in Brazil? A_____

Please answer some questions on the experience you just had with the second user

I found working with the second user frustrating:

Not frustrating at all	a tiny bit frustrating	Slightly frustrating	very frustrating	Highly frustrating, not usable

What effect did the 2nd user have on how long it took to complete the task:

Much slower	Slightly slower	No change	slightly quicker	Much more quickly

Did you have to adapt your actions because the 2nd user was unpredictable? If yes, please describe how below.

- ○ Yes
- ○ No

Please leave a few words to review your experience of working with the second user:

Next

Please fill out these questions

This is task 1 of 6. More questions will slowly appear as you answer the questions.

Which country is the Eiffel Tower located in?	
In which city can you find the Statue of Liberty?	
In which American state can you find Los Angeles?	
Which animal is Wool produced from?	
Which currency is used in the UK?	
Which company makes the Iphone?	
Name the large British city: Ma_____	
In San Francisco you can find the: G_____ G___ Bridge	
The capital of Scotland is: Ed_____	
Complete the sequence: Up Down Left _____	
Complete the sequence: North South East ____	
Which organ of the body pumps blood?	
Where on the body are shoes worn?	
Which organ is responsible for smell	

Please answer some questions on the experience you just had with the second user

I found working with the second user frustrating:

Not frustrating at all	a tiny bit frustrating	Slightly frustrating	very frustrating	Highly frustrating, not usable

What effect did the 2nd user have on how long it took to complete the task:

Much slower	Slightly slower	No change	slightly quicker	Much more quickly

Did you have to adapt your actions because the 2nd user was unpredictable? If yes, please describe how below.

- ○ Yes
- ○ No

Please leave a few words to review your experience of working with the second user:

Next

Section 7 - Ending message

Thank you for participating in our survey. You will now be redirected back to Prolific where your completion will be registered.

Please note that any personal information you may have entered in the survey will be deleted by the researchers.

If you are interested in the results of our study, they will be made available in the coming weeks at `https://killiandavitt.me/collaboration-user-study`

Once again, if you have any questions about the survey please contact us through the Prolific interface. If you wish to email us directly, please do not include any personal information, and in particular do not include your prolific id.

Killian Davitt, [redacted]
Steven Murdoch, [redacted]

Appendix B

HTTPS-Only Mode Scoping Survey Questions

CREWS - HTTPS Upgrading - Tor Project member Survey

This project is investigating the risks associated with users of Tor browser visiting non-HTTPS websites. In particular we are interested in recent developments in browser warning pages, e.g. Firefox's new HTTPS-Only mode which warns users when visiting non-TLS websites. The goal of this survey is to seek broad guidance on what kinds of issues should be discussed if such a mode was to be integrated into Tor Browser. We are not considering Tor Onion Services in this survey, only websites that can be accessed with or without using Tor. This will involve providing some of your opinions on the effectiveness of these modes, what type of action they should be encouraging and how best to provide effective warnings. We are specifically interested in any thoughts you may have on if HTTPS-Only mode should be approached differently in Tor Browser versus other browsers like Firefox.

An invitation to partake in this survey has been extended to members of the tor community in the hopes that their expert advice can inform future work regarding the design of tor browser http warning pages

The data controller for this project will be University College London (UCL). The UCL Data Protection Officer provides oversight of UCL activities involving the processing of personal data, and can be contacted at data-protection@ucl.ac.uk

All data collected in this survey will be subject to GDPR protections and will only be released in a form which cannot identify you. In the case that personally identifiable information is provided, (aside from an identifier which you can optionally consent to providing) this will be manually deleted by the research team before any further processing is conducted

The lawful basis that would be used to process your personal data will be performance of a task in the public interest

The lawful basis that would be used to process any special category data will be scientific or historical research purposes. As this survey does not seek special category data, this lawful basis is only relevant to data which will be deleted by the researchers. I.e. if special category data is entered by participants, it will be deleted before any further processing occurs.

Your personal data will be processed so long as it is required for the research project. If we are able to anonymise or pseudonymise the personal data you provide we will undertake this, and will endeavour to minimise the processing of personal data wherever possible

If you are concerned about how your personal data is being processed, or if you would like to contact us about your rights, please contact UCL in the first instance at data-protection@ucl.ac.uk.

All aspects of this survey are completely optional, you can choose to fill out as much or as little as you wish. You may also withdraw your consent at any time by not submitting the survey or by contacting the researchers and asking for your data to be deleted.

Title of Study: CREWS HTTPS scoping survey
Department: Department of Computer Science, UCL
Name and Contact Details of the Researcher: Killian Davitt, [redacted]
Name and Contact Details of the Principal Researcher: Prof. Steven Murdoch, [redacted]
Name and Contact Details of the UCL Data Protection Officer: data-protection@ucl.ac.uk
This study has been approved by the UCL Research Ethics Committee: Project ID number: 20345/001

Consent

☐ I understand that my participation is voluntary and that I am free to withdraw at any time without giving a reason.

☐ I understand that if I decide to withdraw, any personal data I have provided up to that point will be deleted unless I agree otherwise.

☐ I am aware of who I should contact if I wish to lodge a complaint.

☐ I voluntarily agree to take part in this study.

☐ I confirm that I am over 18 years of age

Acknowledgment

The comments you provide here will inform future work and may contribute to future published work.

☐ I do not wish for my comments to be credited in any final published work

☐ I do wish for my comments to be credited in any final published work

Note: if you have answered/chosen the first answer in the previous question skip the following question

As you have indicated you wish to be credited in further work, please enter your name or identifier which you wish to be credited. Please be aware that the details entered here may be made public in future work

Name:

Questions

Q1: While browsing the web, what broad risks do you think occur for a user visiting non-HTTPS websites?

Q2: Does the level of risk vary depending on the activity being performed? Why do you believe this is the case?

Q3: Is this level of risk greater for users of Tor browser, compared to those connecting directly to the site? Or are there new additional risks to visiting non-HTTPS sites when using Tor browser? What if any, are these risks?

Q4: Can you identify specific criteria which makes visiting a non-HTTPS website more or less risky? In what scenarios are visiting non-HTTPS sites safe?

Q5: For you personally, when faced with a non-HTTPS website, what factors affect whether you will continue to the website, or abandon it?

The goal of this project to attempt to understand how users of Tor browser can understand the risks involved in browsing non-HTTPS websites, and how users can best be informed to make appropriate risk assessments while browsing.

Q6: To this end, can you comment on what technical factors a user must understand in order to make an informed decision regarding non-HTTPS sites being safe or less risky, and why do you think these are important?

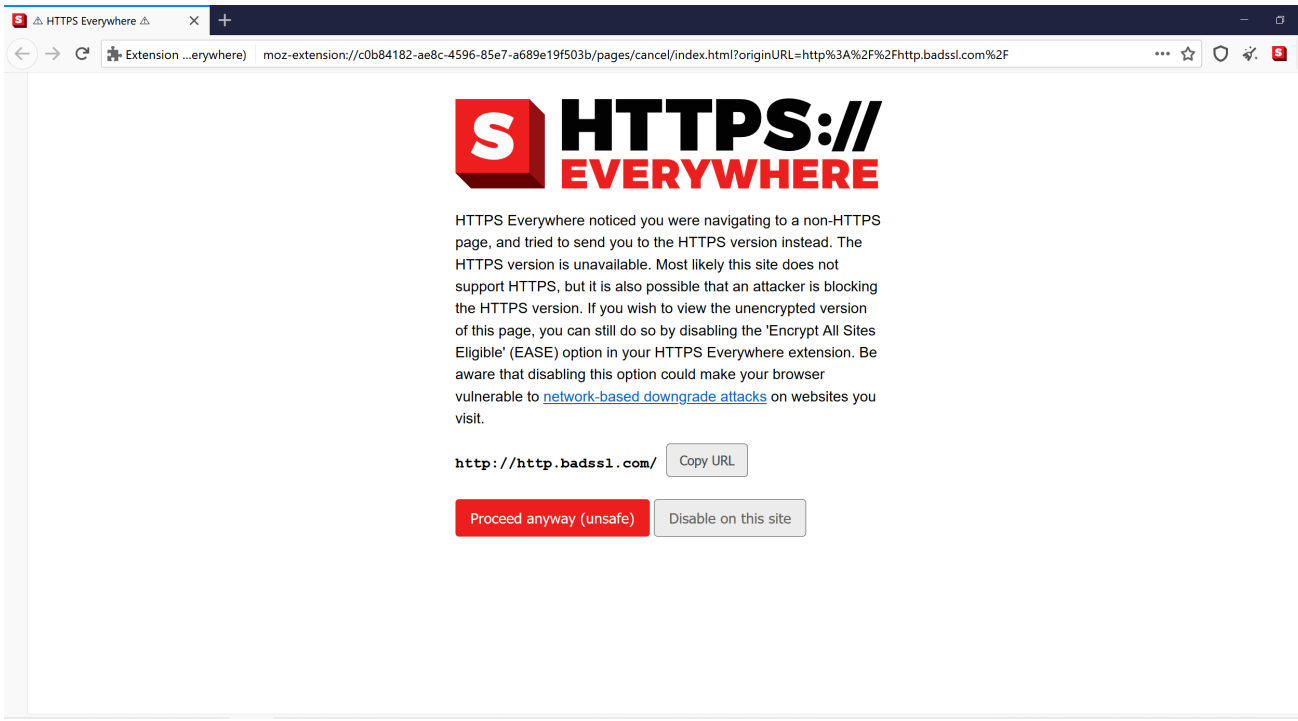
Q7: TLS provides confidentiality of communications, as well as authenticity and integrity. Do you think these concepts, and how they are provided by TLS is well understood by typical Tor browser users?

Q8: Does the level of risk vary depending on the activity being performed? Why do you believe this is the case?

Q9: Do you believe it is necessary to understand confidentiality, authenticity and integrity of HTTP communications in order to make informed decisions about visiting non-HTTPS websites over Tor browser? Why do you think this is the case?

HTTPS Everywhere 'EASE' mode

The following is a screenshot from Tor browser with the HTTPS everywhere 'EASE mode' (Encrypt All Sites Eligible) setting. This is the warning page shown when visiting a non-HTTPS website



Q10: Do you believe this page accurately informs the user of the risks involved in continuing to a non-HTTPS webpage?

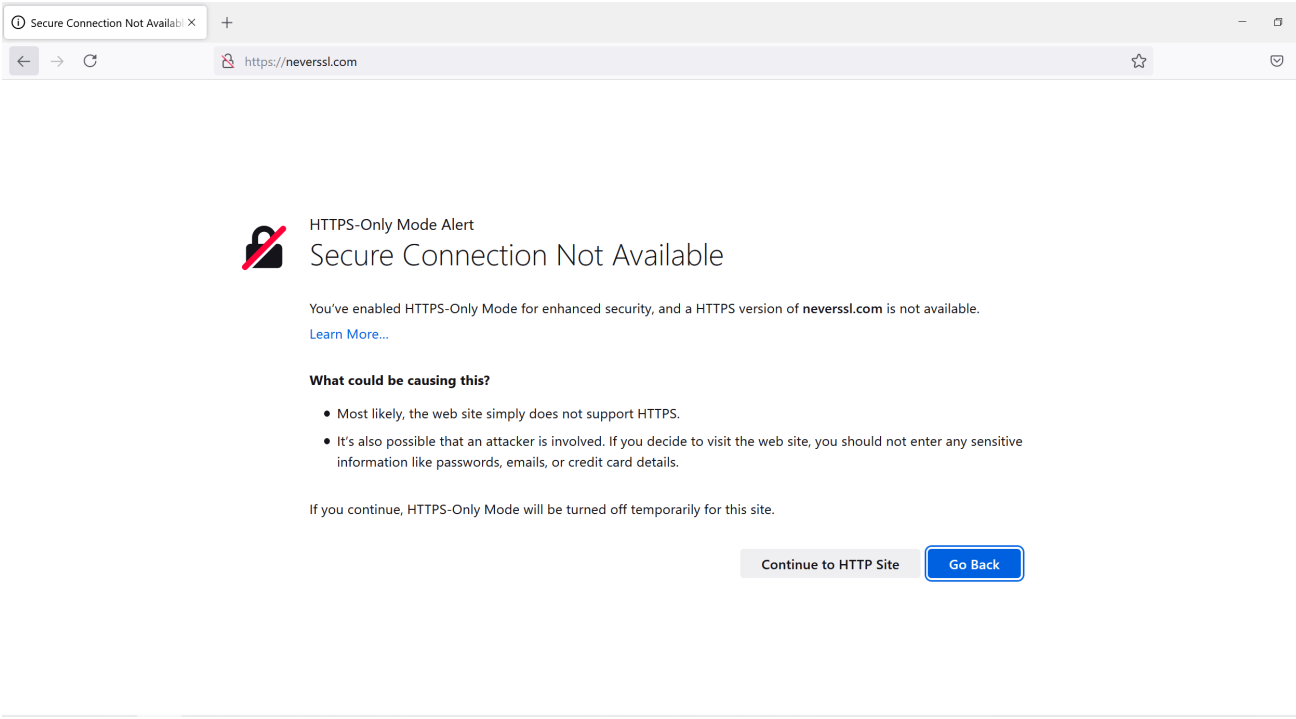
Q11: If not, what issues are left out of this page?

Q12: Do you believe this page overstates the dangers of non-HTTPS websites?

Firefox

Firefox

The following is a screenshot from the new firefox 'HTTPS only' setting. This is the warning page shown when visiting a non-HTTPS website



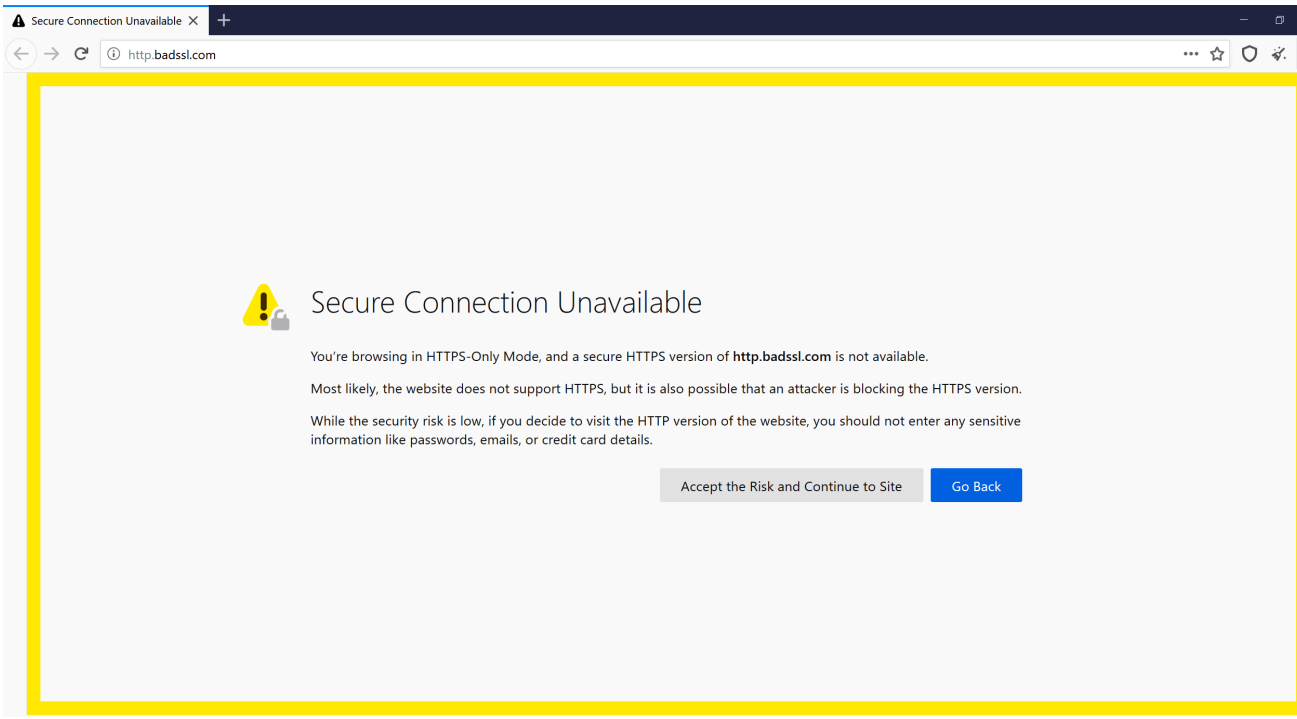
Q13: Do you believe this page accurately informs the user of the risks involved in continuing to a non-HTTPS webpage?

Q14: If not, what issues are left out of this page?

Q15: Do you believe this page overstates the dangers of non-HTTPS websites?

Tor

The following is a screenshot from the current Tor browser 'HTTPS only' setting. This is the warning page shown when visiting a non-HTTPS website



Q16: Do you believe this page accurately informs the user of the risks involved in continuing to a non-HTTPS webpage?

Q17: If not, what issues are left out of this page?

Q18: Do you believe this page overstates the dangers of non-HTTPS websites?

Post Survey Message

Thank you for taking our survey. If you are interested in the results of our survey, they will be made available in the coming weeks at `https://murdoch.is/projects/#crews-hdi` If you have any further questions please contact:

Killian Davitt, killian.davitt.17@ucl.ac.uk

Prof. Steven Murdoch, s.murdoch@ucl.ac.uk

Recruitment Email

To: tor-dev@lists.torproject.org Subject: Survey on HTTPS-Only Mode in Tor Browser Hello All,

We are currently recruiting participants for a short online survey regarding HTTPS-Only modes in Tor Browser. This will involve providing some of your opinions on the effectiveness of these modes, what type of action they should be encouraging and how best to provide effective warnings. We are specifically interested in any thoughts you may have on if HTTPS-Only mode should be approached differently in Tor Browser versus other browsers like Firefox.

This is part of the CREWS - HCI project which is a collaboration between University College London and The Tor Project. The overall goal of our project is to construct an effective and well researched HTTPS-Only mode for Tor, that can potentially be enabled by default

If you are interested in participating, more details can be found here: `https://opinio.ucl.ac.uk/s?s=73764` Our survey is totally anonymous unless you wish to give your name to be credited in future work. The survey will be closed on [TBC].

If you have any thoughts whatsoever about the future of HTTPS-Only mode in Tor Browser, then please feel free to reply to this email with them also.

This study has been approved by the UCL Research Ethics Committee: Project ID number: 20345/001

Thank you, Killian Davitt & Steven Murdoch
https://killiandavitt.me/ [redacted]
https://murdoch.is/ [redacted]

Appendix C

Evaluating New Warning Pages Survey Questions

Section 1: Information & Consent

This goal of this project is to improve "HTTPS Only Modes" in the Tor Browser web browser. HTTPS Only Modes are modes which warn users when a website does not have HTTPS which is an important security feature. Visiting a website without HTTPS is only risky in some circumstances, and we have designed warning pages which attempt to properly convey this risk.

You will be asked what action you would take in various web browsing scenarios, when presented with one of our warning pages. You will only see 1 warning page for the duration of this survey. You will have been recruited for this survey through the Prolific platform, and you will receive your payment from Prolific once you complete the survey.

The data controller for this project will be University College London (UCL). The UCL Data Protection Officer provides oversight of UCL activities involving the processing of personal data, and can be contacted at data-protection@ucl.ac.uk

As a result of your participation in this survey, we will receive your demographic data which you have previously provided to Prolific. We will also ask you for your approximate education level. The time taken for you to complete each question will also be recorded. All data collected in this survey will be subject to GDPR protections and will only be released in a form which cannot identify you.

The lawful basis that will be used to process your personal data will be performance of a task in the public interest. We do not seek any personal data in this survey, and you should not enter personal data into any of the textboxes.

The lawful basis that would be used to process any special category data will be scientific or historical research purposes. As this survey does not seek special cat-

egory data, this lawful basis is only relevant to data which will be deleted by the researchers. I.e. if special category data is entered by participants, it will be deleted before any further processing.

If at any point you enter personal data in this survey, it will be removed by the researchers before any other processing occurs. Additionally, if you disclose your Prolific ID to the researchers through any communications, it will be deleted immediately in order to not deanonymise you.

Unless you enter personal data into the text boxes, we will only receive data from you that is anonymous. If you are concerned about how your data is being processed, or if you would like to contact us about your rights, please contact UCL in the first instance at data-protection@ucl.ac.uk.

Please See UCL's General privacy notice for more information on UCL's data protection policies `https://www.ucl.ac.uk/legal-services/privacy/ucl-general-research-participant-privacy-notice`

If you wish to revoke your consent for the processing of your data, you can revoke consent through the Prolific platform. Please note that you may only withdraw your consent 7 days after you submit your response, as the data collected will be anonymous, there will be no way to delete you data after the survey closes.

If you decide to email the researchers directly, please do not quote your prolific id, as this will deanonymise you. If you include your prolific id in your email to us, we will delete the email without responding in order to not deanonymise your data.

Title of Study: Improving HTTPS Only Mode Warning Pages
Department: Department of Computer Science, UCL
Researcher: Killian Davitt, [redacted]
Principal Researcher: Prof. Steven Murdoch, [redacted]
UCL Data Protection Officer: data-protection@ucl.ac.uk

This study has been approved by the UCL Computer Science Research Ethics Committee: UCL/CSREC/R/10

- ☐ I understand that my participation is voluntary and that I am free to withdraw at any time without giving a reason
- ☐ I understand that my data gathered in this study will be stored anonymously and securely. It will not be possible to identify me in any publications.
- ☐ I am aware that I can only withdraw my data up to 7 days after my submission, and this can be done only through the Prolific user interface
- ☐ I am aware of who I should contact if I wish to lodge a complaint
- ☐ I voluntarily agree to take part in this study.
- ☐ I confirm that I am over 18 years of age

Section 2

Please enter your current highest level of formal education

- ☐ None, or incomplete secondary education
- ☐ Secondary education (A levels, GCSE, further education, etc)
- ☐ University degree (BA, BSc, etc)
- ☐ Graduate degree (Masters, PhD, etc)

Have you used HTTPS Only Modes before? (Note that this is a security feature which you would have had to deliberately enable in your browser settings. If you have not heard of this, it is very unlikely you have used it before)

- ☐ Yes
- ☐ No

Section 3

You will now be presented with a series of hypothetical web browsing scenarios and shown the browser page which appears. You will be asked what action you believe you would take. Please do not enter any personal information into any of the text boxes. This includes, names, email addresses, or any identifier which could deanonymise your submission.

In all the following questions, imagine you are staying at a hotel. Upon check-in the receptionist notifies you that the regular hotel WiFi is broken. They also inform you that another unnamed guest has offered to share their own WiFi connection with other guests and it should be available under the name: "Free WiFi". You connect to this network and continue to browse the web.

For the following questions, how were you asked to imagine you are connecting to the internet? Required

☐ An unsecured network you find at a hotel

☐ An unsecured network you find at the airport

☐ Your home internet connection

☐ A Wi-Fi connection provided by a local cafe

(An incorrect answer to this question results in the participant being directed to section 4, otherwise participants are directed to section 6)

Section 4

(Participants are directed to this section if they answer incorrectly the previous question)

Please read these instructions carefully and attempt to answer the question again

In all the following questions, imagine you are staying at a hotel. Upon check-in the receptionist notifies you that the regular hotel WiFi is broken. They also inform you that another unnamed guest has offered to share their own WiFi connection with other guests and it should be available under the name: "Free WiFi". You connect to this network and continue to browse the web.

For the following questions, how were you asked to imagine you are connecting to the internet?

☐ An unsecured network you find at the airport

☐ A Wi-Fi connection provided by a local cafe owner

☐ A WiFi connection at a hotel provided by another anonymous guest

☐ Your home internet connection

(If the participant answers this question incorrectly again, they are not allowed to participate in survey and they are directed to section 5. If they an-

swer correctly, they are directed to section 6.)

Section 5

(Participants are directed to this section if they answer incorrectly the previous question)

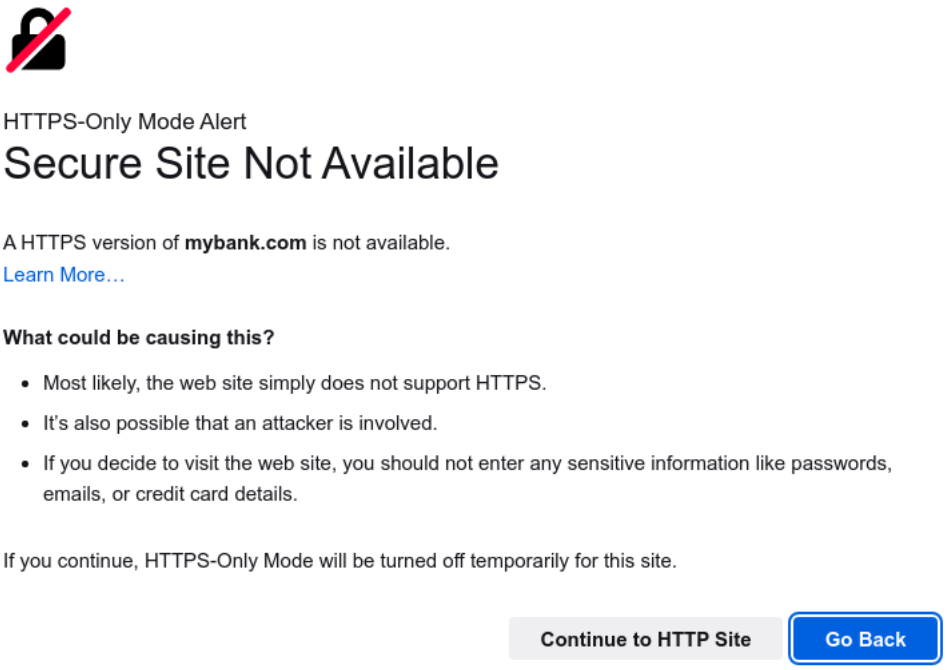
Unfortunately you have not demonstrated a thorough understand of the question being asked. Please close the survey and return to Prolific and select 'Stop Without Completing'.

– Survey ends here –

Section 6 - Main Questions

Scenario 1

In this scenario, imagine you are browsing the web, and you decide to check your bank account balance. You proceed to your bank's website, and are presented with this warning.



1. What action do you take?

   ☐ Press the continue button

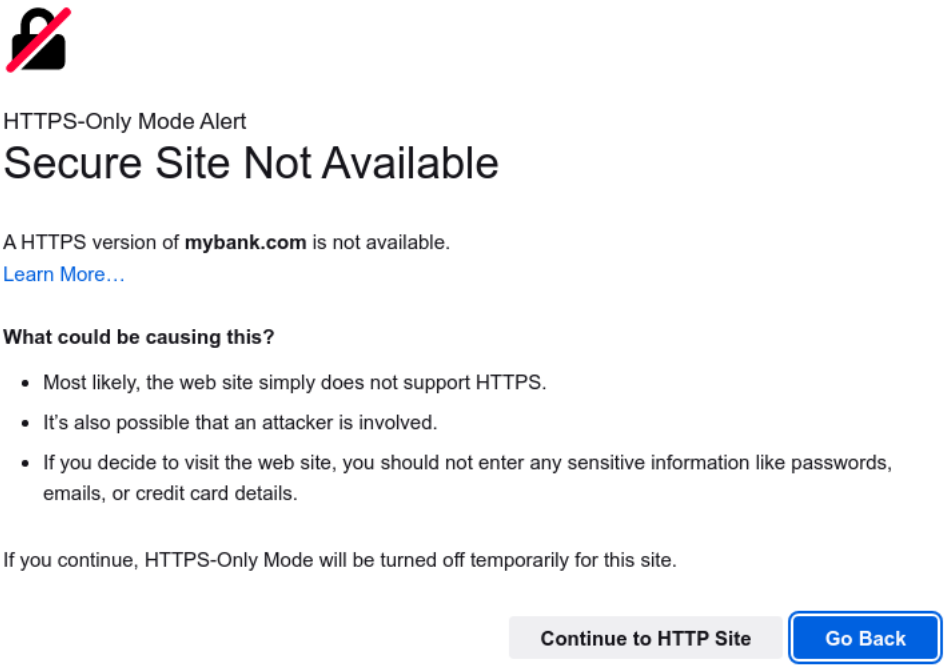
   ☐ Click "Go Back" or close the browser tab

   ☐ Other (please specify) _____

2. Why did you take this action?

Scenario 2

In this scenario, image you are browsing the web, a friend sends you a link to what looks like a news website, you have not heard of the website before. You click the link, and are presented with this warning.



1. What action do you take?

   ☐ Press the continue button

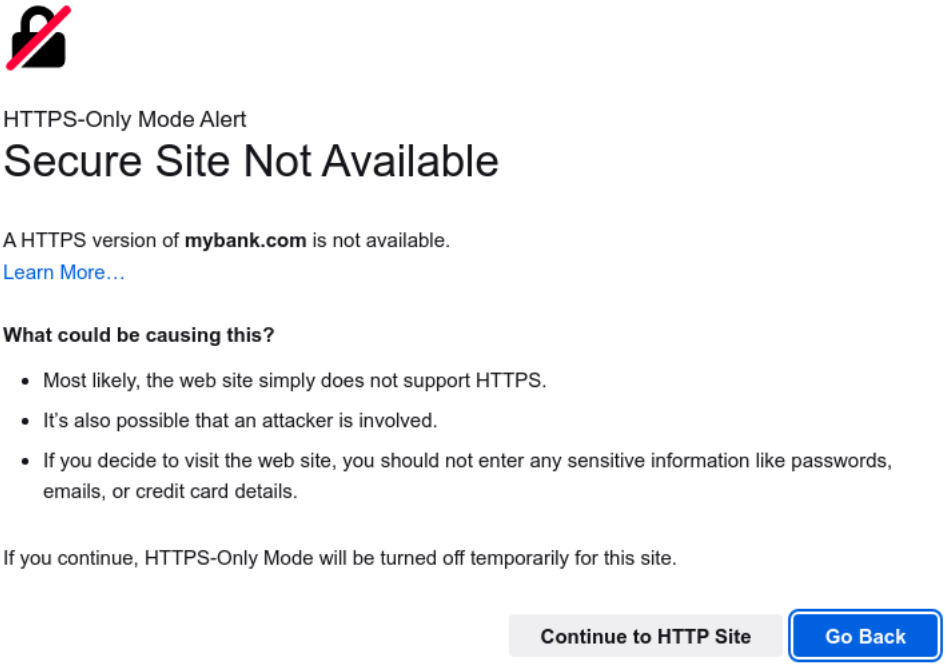
   ☐ Click "Go Back" or close the browser tab

   ☐ Other (please specify) _____

2. Why did you take this action?

Scenario 3

In this scenario, you visit a well known news site to see the latest news, you are then presented with this warning.



1. What action do you take?

   ☐ Press the continue button

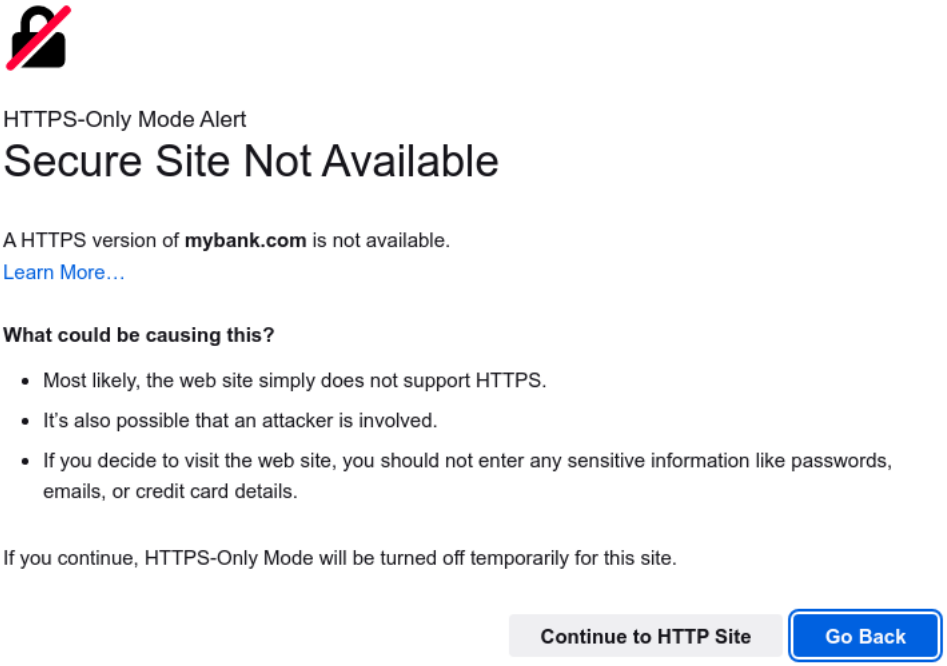
   ☐ Click "Go Back" or close the browser tab

   ☐ Other (please specify) _____

2. Why did you take this action?

Scenario 4

In this scenario, you want to find the menu for a small local restaurant, after searching the web for their website, you are presented with this warning.



1. What action do you take?

   ☐ Press the continue button

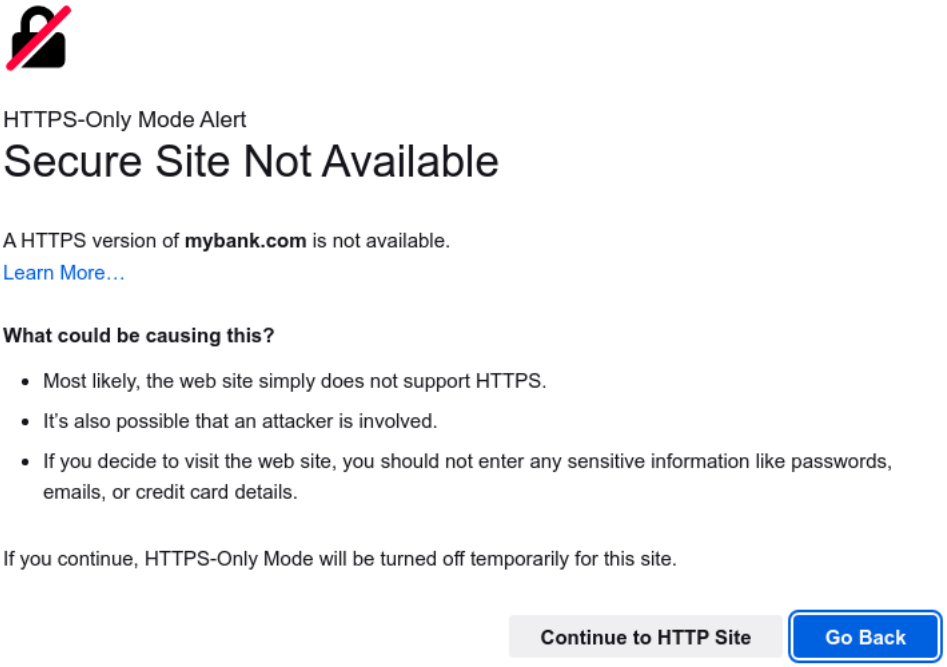
   ☐ Click "Go Back" or close the browser tab

   ☐ Other (please specify) _____

2. Why did you take this action?

Scenario 5

In this scenario, you are buying a gift for someone through a small independent shop, you browse their website hoping to make the purchase online, but you are presented with this warning.



1. What action do you take?

   ☐ Press the continue button

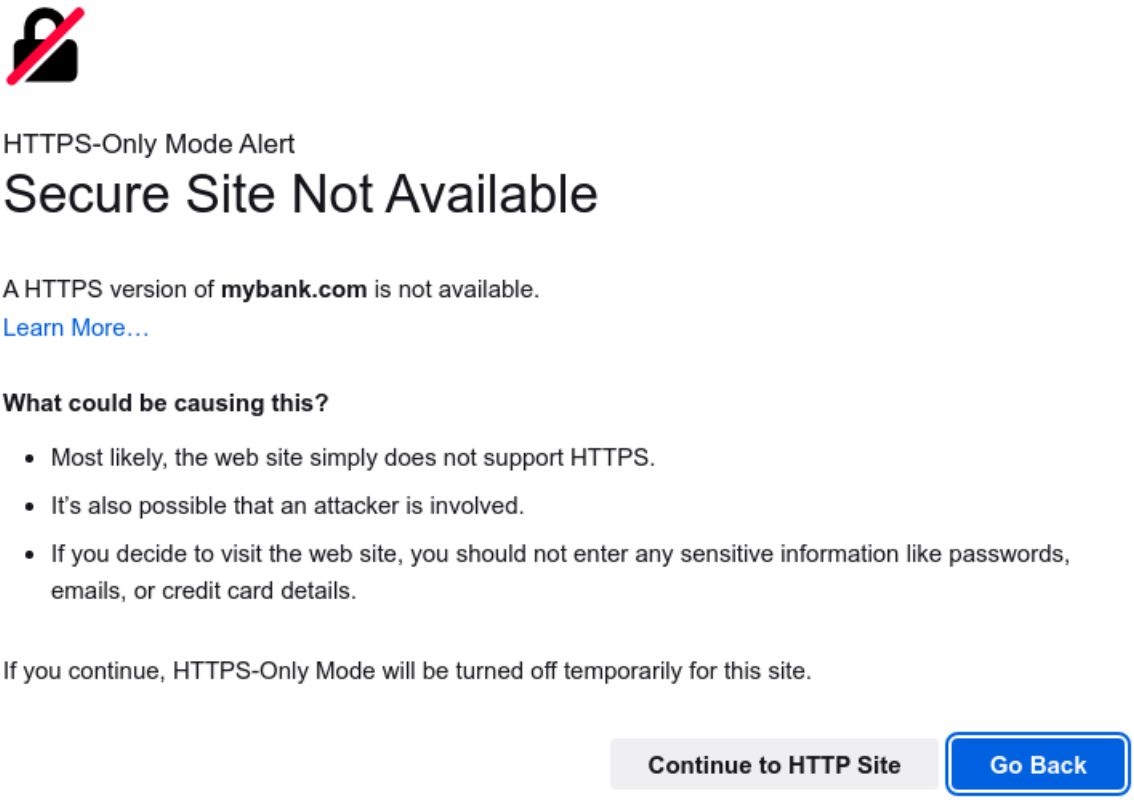
   ☐ Click "Go Back" or close the browser tab

   ☐ Other (please specify) _____

2. Why did you take this action?

Scenario 6

In this scenario, you are buying an item from a large, reputable and popular online retailer, you are presented with this warning



1. What action do you take?

   ☐ Press the continue button

   ☐ Click “Go Back" or close the browser tab

   ☐ Other (please specify)

2. Why did you take this action?

   [ ]

Section 7 - Other Images

(For each participant, one of the following images is selected to replace the images in the 6 previous scenarios, such that an equal proportion of participants will see each of the 4 total images.)

Image 2

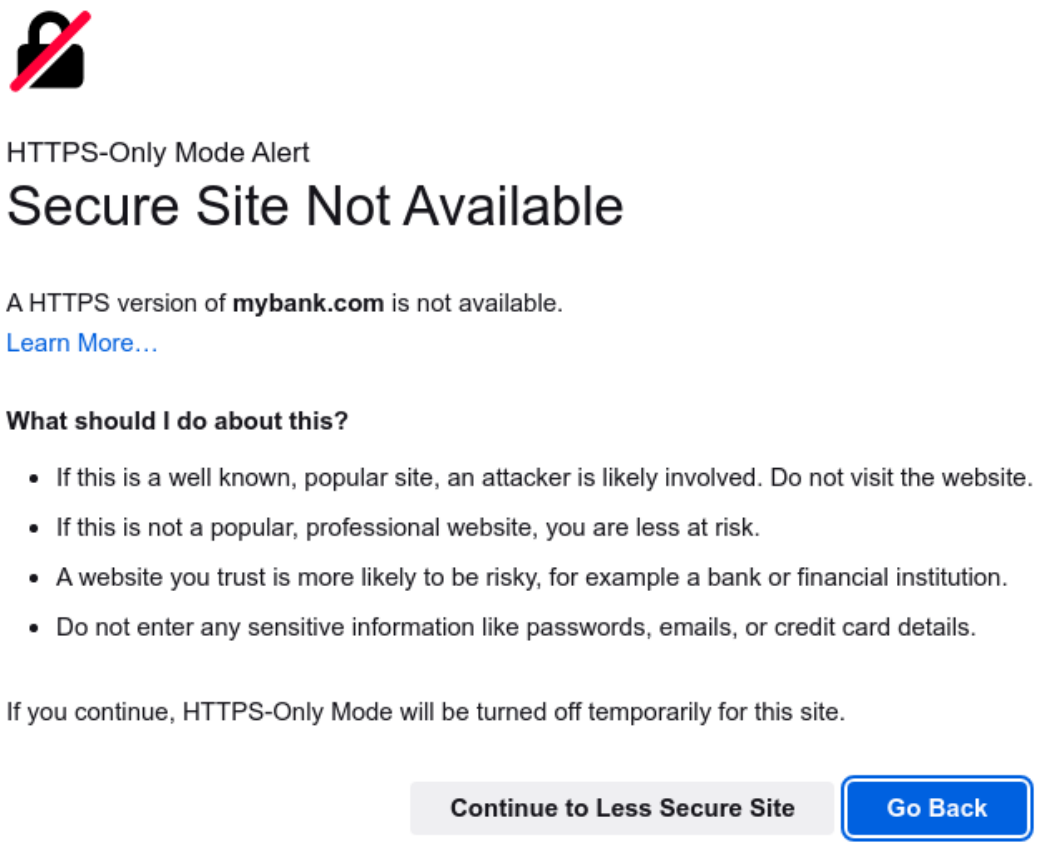


Image 3

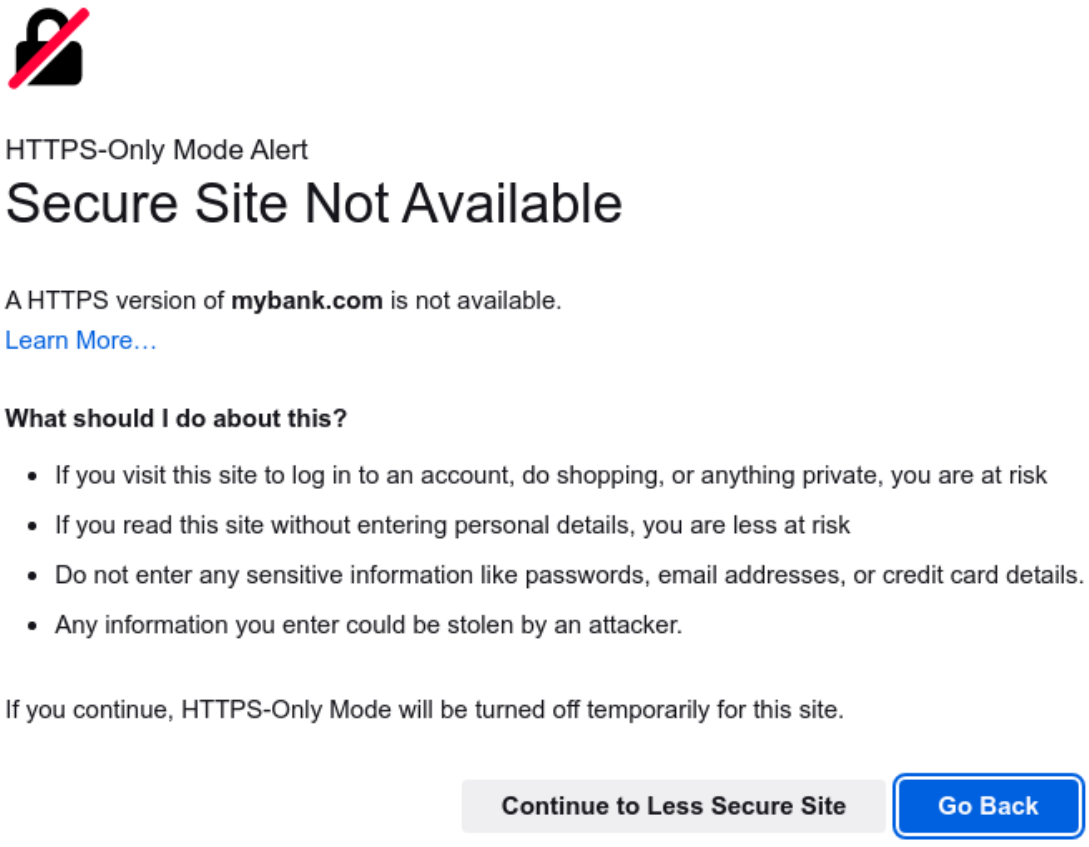
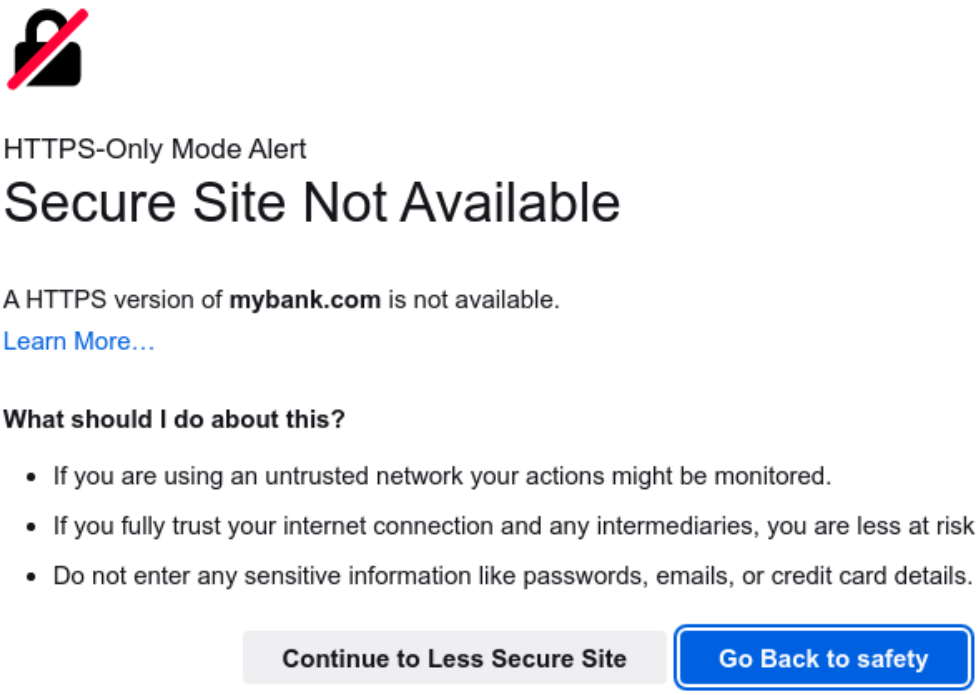

HTTPS-Only Mode Alert
Secure Site Not Available
A HTTPS version of mybank.com is not available.
Learn More…
What should I do about this?
If you visit this site to log in to an account, do shopping, or anything private, you are at risk
If you read this site without entering personal details, you are less at risk
Do not enter any sensitive information like passwords, email addresses, or credit card details.
Any information you enter could be stolen by an attacker.
If you continue, HTTPS-Only Mode will be turned off temporarily for this site.
Continue to Less Secure Site
Go Back

Image 4



Other text on images

Each image can list the name of the website as any of the following:

- mybank.com
- deliciouspizza.com
- homemadegifts.com
- retailer.com
- importantnews.com
- localnews.com

Section 8 - Ending message

Thank you for participating in our survey. You will now be redirected back to Prolific where your completion will be registered.

Please note that any personal information you may have entered in the survey will be deleted by the researchers.

We would also now like to provide some explanation for what the ideal responses to the scenarios presented were.

Firstly you are asked to imagine browsing the web using an unknown WiFi connection at a hotel. This is designed to simulate the use of a web browser called Tor Browser which provides anonymity to its users. Due to the design of Tor, its users have their traffic routed through unknown third parties. This can make them more vulnerable to connection monitoring if they are not visiting a website that employs HTTPS. We asked you to imagine that your connection was also being routed through this unknown individual providing WiFi at the hotel, as this puts you under a similar risk to those using Tor Browser. This will allow us to apply our results to Tor Browser specifically.

In the scenarios you were given, to evaluate which action is best, we need to understand what the purpose of HTTPS is. The most important purpose of HTTPS is to encrypt your data. This means that if you do not have HTTPS it is possible for a third party attacker to see all of the information you transmit to this website.

In some scenarios this is not an issue; if you are browsing the news for example you are likely less worried about an attacker seeing which articles you are reading. On the other hand, if you are doing your banking, or typing your credit card details on a website, you most certainly do not want an attacker seeing this information.

The second major factor to consider is which websites usually have HTTPS. At present, HTTPS is very widely used. Any major company, bank, retailer, service will be highly likely to have enabled this feature. It is a gold standard security

protection, and so, it is effectively a major red flag if a popular website does not have HTTPS available. If a major website like the BBC for example does not appear to have HTTPS enabled, it is highly likely that an attacker has intercepted your connection, and that you are in fact visiting the attackers website instead of the BBC. Conversely, if you visit the website of a small local restaurant, they may not have the resources or time required to setup HTTPS and so it is not alarming that they might not provide it. It is however still dangerous to enter personal detail, account info, or banking details on any website which does not provide HTTPS, regardless of the website popularity.

If you are interested in the results of our survey, they will be made available in the coming weeks at `https://murdoch.is/projects/#crews-hdi`

Once again, if you have any questions about the survey please contact us through the Prolific interface. If you wish to email us directly, please do not include any personal information, and in particular do not include your prolific id.

Killian Davitt, [redacted]
Steven Murdoch, [redacted]