TU WIEN Informatics

Measuring Privacy-Enhancing Technologies

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor der Technischen Wissenschaften

by

Dipl.-Ing. Wilfried Mayer, BSc.
Registration Number 00728092

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.-Prof. Dipl.-Ing. Mag. Dr. techn. Edgar Weippl

The dissertation has been reviewed by:

Dominik Engel	Günther Pernul

Vienna, 30th October, 2021

Wilfried Mayer

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Dipl.-Ing. Wilfried Mayer, BSc.

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 30. Oktober 2021

Wilfried Mayer

Acknowledgements

This thesis would not have been possible without the support of many people.

I'm incredibly grateful to my advisor, Edgar Weippl. He supported me throughout this endeavor and provided me with the freedom to pursue my ideas and work on this thesis.

I would also like to extend my deepest gratitude to Martin Schmiedecker for guiding me through the first stages of academia and providing me with encouragement and patience throughout this project. Special thanks to my fellow researchers: To Georg Merzdovnik for plenty of discussions, constructive criticism, and especially for supporting me in the final phase of this thesis. And to Katharina Krombholz and Adrian Dabrowski. Working on research projects together cultivated my scientific mindset. I also had the great pleasure of working with Gabriel Gegenhuber. His initiative and his ideas are remarkable. I'd also like to acknowledge all other co-authors of my publications: Aaron Zauner, Thomas Schreiber, and Markus Donko-Huber. And of course, I also thank my research center SBA Research for providing me with the environment to work on my dissertation.

Furthermore, I thank all my friends – particularly Tobias – for their moral support all these years. And I thank Franziska for cheering me up in the final phase of this thesis.

Last, I want to thank my parents Sigrun and Erwin, and my sister Nina for supporting me throughout my life. You awoke my curiosity when I was young and encouraged it all the time.

Abstract

The digitalization of all parts of our lives is ongoing. Apart from the positive aspects of digitalization, the momentum of this progression is also disrupting current, well-defined practices. These practices include broad areas like jurisdiction, parts of the economy, and sociological and democratic practices, including privacy. Although privacy is and stays a fundamental human right, the way we perceive, honor, and live with this right has changed. Additionally, the amount of information that is handled by technical systems has increased considerably. Different communities approached this problem from diverse – including legal, sociological, and philosophical – vantage points. The technical community responded with privacy-enhancing technologies (PETs), which provide the same functionality but reduce necessary data to enhance privacy. These systems are vital tools to enhance users' privacy, but to what extent is often unclear. Therefore, we need measurement systems to study the impact of privacy-enhancing technologies.

This thesis improves online privacy from a technical vantage point by measuring PETs. This is accomplished by advancing the state of the art by designing and implementing measurement systems to examine large-scale or long-term measurements. We conduct measurements, combine different approaches and techniques, and overcome technical, organizational, and ethical challenges. Afterwards, we empirically evaluate the results and derive parameters of interest. In each step, we strive to practice openness.

This thesis addresses different areas. First, we analyze the security and privacy of data in transit, which is most commonly achieved with Transport Layer Security (TLS). The capabilities of TLS are extensive, and it can be seen as the fundament of today's web security. However, research gaps still exist. Consequently, we summarize and extend the results of our large-scale study of TLS in Non-HTTP settings. Further, we develop and evaluate new approaches for cipher suite scanning, which is an important tool to evaluate the current status of the TLS ecosystem. We then describe the long and challenging process of fully adopting HTTPS and showed multiple ways to support it. Last, we introduce a new approach that generates client-side rewrite rules automatically.

Second, we consider potential middlemen that could infer private information from metadata. Because threats from malicious actors are highly studied, we focus on so-called trusted parties, like Internet service providers. We monitor techniques from different Internet providers with our long-term net neutrality study, including middleboxes and methods used for blocked and non-existing DNS requests. We also solve the scalability

problem for cellular network measurement frameworks. A low entry threshold for unique measurement opportunities in a specific country or internationally is created with our SIM decoupling method.

Third, we analyze technical systems that let users protect their metadata, in particular, anonymity systems. Therefore, we present a novel way to analyze the network routes taken by traffic from and to the Tor network, the most prominent anonymity system online. We extend previous research that relies on the analysis of BGP routing information and simulations and utilize the RIPE Atlas framework to measure network routes.

With this thesis, we refine existing measurement methods. We define, implement, deploy, and evaluate novel measurement frameworks. We conduct measurements to indicate the current state, and we gain insights into the causes of poor PETs deployment quality.

Kurzfassung

Die Digitalisierung aller unserer Lebensbereiche ist unaufhaltbar. Neben den positiven Aspekten der Digitalisierung verändert dieser Fortschritt aber auch aktuell klar definierte und akzeptierte Strukturen. Dies betrifft Rechtsprechung, Bereiche der Wirtschaft, Vorgänge in der Gesellschaft und unsere demokratische Gesellschaftsordnung. Es verändert auch unser Konzept von Privatsphäre. Obwohl Privatsphäre ein Grundrecht darstellt, ändert sich die Art wie wir Privatsphäre wahrnehmen, würdigen und wie wir unser Recht auf Privatsphäre ausüben. Zusätzlich dazu wächst die Menge an sensibler Information, die durch technische Systeme verarbeitet wird beträchtlich. Es gibt verschiedene Zugangsweisen, um mit diesem Prozess umzugehen. Das sind unter anderem rechtliche, soziologische oder philosophische Aspekte. Technische Annäherungen sind „Privacy-enhancing technologies“(PETS) – datenschutzfördernde Technologien. Diese stellen Funktionalität bereit, reduzieren dabei aber die verwendeten Daten. PETS sind wichtige Werkzeuge um die Privatsphäre von Benutzern*innen zu stärken, aber in welchem Ausmaß ist oft unklar.

In dieser Dissertation wird die Privatsphäre von Benutzer*innen von einem technischen Standpunkt aus gestärkt. Dies erfolgt durch die Verbesserung von Messungssystemen, durch Entwicklung und Implementierung von Messungssystemen und durch die Durchführung von groß angelegten und langzeitigen Messungen im Umfeld von PETS. Dabei werden Messungen durchgeführt, verschiedene Herangehensweisen und Techniken kombiniert und technische, organisatorische und ethische Herausforderungen bewältigt. Anschließend werden Ergebnisse evaluiert und Kennwerte davon abgeleitet. Bei jedem Schritt wird Transparenz angestrebt.

In dieser Arbeit werden verschiedenen Bereiche von PETS behandelt. Es wird die Sicherheit und Privatsphäre von Daten auf dem Transportweg analysiert. Hier wird üblicherweise Transport Layer Security (TLS) eingesetzt. Die Möglichkeiten von TLS sind umfangreich und es kann als das Fundament heutiger Web-Sicherheit gesehen werden. Es werden die Ergebnisse einer groß angelegten Studie von TLS in Nicht-HTTP Konfigurationen resümiert und erweitert. Neue Ansätze für Cipher-Suite Scanning werden entwickelt und evaluiert, da dies ein wichtiges Werkzeug darstellt, um den aktuellen Status von TLS-Konfigurationen zu messen. Außerdem wird auf den herausfordernden Prozess eingegangen HTTPS zum neuen Standard im Web zu forcieren. Eine crowdbasierte Technik um clientseitige Umschreiberegeln für HTTPS zu generieren schließt die TLS-Analyse ab.

Entitäten am Übertragungsweg können jedoch trotzdem private Informationen von Metadaten ableiten. Die Arbeit fokussiert sich dabei auf sogenannte vertrauenswürdige Entitäten, nämlich Internet Service Provider. Bei einer langfristigen Netzneutralitätsstudie werden Techniken, wie Middleboxes und verschiedenes DNS Verhalten beobachtet. Außerdem wird mit einem neuen Ansatz das Problem der Skalierbarkeit von Messungen in mobilen Netzen gelöst. Durch unsere SIM-Entkopplungstechnik wird eine geringe Eintrittshürde für weitere Messungen erreicht.

Schlussendlich können technische Systeme wie Anonymitätssysteme Metadaten schützen. Diese Arbeit präsentiert einen innovativen Weg Netzwerkrouten vom und zum Tor Netzwerk zu analysieren. Dabei wird das RIPE Atlas Netzwerk genutzt um Netzwerk-Routen zu messen. Dies erweitert vorhergehende Forschung, welche auf den Analysen von BGP-Routeninformation und Simulationen beruht.

Mit dieser Arbeit werden bestehende Messmethoden verbessert. Neuartige Messungssysteme werden definiert, implementiert und evaluiert. Der Ist-Zustand wird mit Messungen erhoben und Erkenntnisse über die Ursachen für eine mangelhafte Qualität von eingesetzten PETS werden gewonnen.

Contents

CHAPTER 1

Introduction

The ongoing digitalization is affecting all parts of our lives. This steadily changes our perception of privacy. We abandon personal data to computer systems, often unaware of possible consequences. To which extend this process affects us individuals is often uncertain. Privacy-enhancing technologies (PETs) are a technological way to deal with this situation. To clarify this situation, we formulate a problem statement for measuring PETs online in Section 1.2. Technologies often vary and interact within complex ecosystems, so special care has to be taken to quantify the parameters of these systems. The main results of this thesis are already published in various peer-reviewed publications. We summarize the results and end with a list of publications related to this thesis in Section 1.4.

1.1 Motivation

In the last decades, nearly all aspects of our daily lives have been digitalized. Even domains that were not digitalized one decade ago changed completely. This development is unstoppable, and we expect even more elements of our lives to be digitalized in the future. With the ongoing COVID-19 pandemic and the concomitant physical distancing rules, we also witness a significant event that will transform even more domains to the digital world. This progression clearly has positive aspects, as it opens up a lot of opportunities for our society and, ultimately, our personal lives. Nevertheless, the momentum of this progression is disrupting current, well-defined practices. This includes broad areas like jurisdiction, parts of our economy, as well as personal interactions. Also, long-established rules of privacy are disrupted, e.g., the privacy of correspondence. In an analog world, this was often seen as an indisputable right. Digitally, however, it was neglected for a long time, discussed on various bases, or even announced depreciated as in Zuckerberg's comment that privacy is no longer a social norm [1]. With the Snowden revelations in 2013, we even witnessed the scale of the abuse of privacy rights of countless individuals in a global surveillance system. Still, privacy is and stays a fundamental

human right as declared in the *Universal Declaration of Human Rights*[1]. Discussions about this topic are often addressed from diverse vantage points. This includes legal, sociological, and philosophical approaches. However, in this thesis, we concentrate on a technical viewpoint of individual privacy. This is relevant, because the technical community responded to these discussions with the introduction of many so-called PETs. PETs are ways to protect privacy online by eliminating or minimizing personal data without the loss of functionality of the technology [2]. These tools are the technological means to protect the human right to privacy in this digitalized world. The definition of PETs seems to be pretty abstract, but by applying the included concepts to the online behavior of individual users, we can see that the individuals' privacy is often protected by PETs. And because digitalization pervades *all aspects* of our lives, the question arises which aspects are considered sensitive and deserve strong protection. We immediately find common sensitive topics, e.g., financial data, medical and psychological data and treatments, societal decisions, and involvement in democratic processes. When we look at it from a technical viewpoint, we identify PETs protecting the users' privacy from different technical directions. Either data is protected in rest, i.e., stored somewhere, with, e.g., encryption or data anonymization approaches. Alternatively, data is protected while in transit, moving between different computers and administrative entities, completely independent from the originating user. Does the – already mentioned – comparison to the analog world hold for this type of data transmission?

The users' communication is under a pervasive privacy threat. How big this threat is and how mentioned PETs prevent or affect this threat is often unclear. Only with measurement systems, we would have an efficient way to scientifically show the usefulness of different approaches. And although some measurement scenarios exist, there is plenty of room for improvement.

1.2 Problem Statement

This thesis improves privacy online and strengthens digital self-determination by providing and improving techniques to measure ecosystems of PETs affecting the individual user.

Privacy in transport As users communicate and transmit data over the Internet, they use different technologies. They interact with a plethora of different applications, and a comprehensive analysis of all applications is just not feasible. However, many of these applications use a common denominator for transmitting data, namely Transport Layer Security (TLS). TLS is the most used privacy-enhancing technology and is today's foundation of privacy online. With such importance and widespread usage, the TLS ecosystem has multiple dimensions to be considered. Because each dimension can break the security properties of TLS and ultimately undermine the users' privacy, there is a

[1] As declared in the *Universal Declaration of Human Rights, Article 12*: "No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honor and reputation.

need for further research. This thesis researches the adoption in different application ecosystems, the quality of used cryptographic primitives, and the way experts are setting up trusted environments.

Privacy and network providers Although TLS improves the users' privacy, it does not protect metadata from a potential middleman. These middlemen know the timing, source, and origin of data. Also, additional information could be leaked, e.g., through the Server Name Indication (SNI) in the TLS handshake. While solutions to tackle some of these problems currently emerge, e.g., Encrypted Server Name Indication (ESNI), these technologies are not yet widely deployed. The user is often unaware of such threats, and even with knowledge, so-called trusted third parties, like Internet Service Providers (ISPs), could be potential malicious actors. These ISPs are often strongly regulated, which makes evading users' privacy appear unlikely. However, they use techniques for network management and traffic differentiation. Used metadata analysis techniques are often relevant for the area of Net Neutrality. Nevertheless, the same methods could be used for metadata analysis. The second part of this thesis gives a closer look at these practices. It covers practices of regular ISPs as well as mobile providers.

Privacy in metadata-free communication Nevertheless, non-trivial solutions to circumvent this metadata-leaking transmission exist. Users could opt for anonymity networks to hide their metadata. While many different academic anonymity systems exist, only a few are adopted by a considerable user base. The Tor project is the most widely adopted system. Although not providing information theoretical anonymity, with its onion routing architecture it is efficiently protecting Internet Protocol (IP) address leakage. Furthermore, other parts of the project like the Tor Browser hide other sensible information and protect the user on a larger scope. Still, as an overlay network, it hardly relies on the underlying network. The third part of this thesis refines methods analyzing routing strategies to strengthen Tor.

1.3 Methodology

This thesis is based on several peer-reviewed publications. These publications mostly follow a multi-step procedure to address the stated problems. A detailed insight is given in the respective sections of this thesis.

- First, a **comprehensive literature review** is conducted. The current state of the art is evaluated by reviewing published literature. It also analyzes short-lived, non-peer-reviewed approaches, open-source measurement methods, existing scanning infrastructures, and existing datasets.
- Second, a **measurement design** and a **measurement implementation** are created. These systems have to examine large-scale measurements and incorporate non-functional requirements, e.g., openness and scalability.

- Third, **active and passive scans** are conducted. These scans combine newly created measurement approaches and existing techniques. However, large-scale scanning imposes technical challenges that have to be solved. Additionally to the technical challenges, also ethical and organizational requirements have to be addressed.

- Last, we **empirically evaluate the results** of the measurements. Here, we quantify specific parameters of relevance for the healthiness of the measured privacy systems. This is done with the consolidation of other publicly available datasets.

- Further, we strive to **practice openness**. Next to publishing academic publications, we release the source code open-source, the obtained data sets, and the evaluation scripts (see Section 1.4.2). This is especially important to facilitate reproducibility and replicability.

1.4 Main Results

The **first part of this thesis** focuses on **Transport Layer Security**. As described above, the security properties of TLS depend on multiple dimensions (e.g., cryptographic primitives used, deployment quality, usability). In this thesis, we analyze multiple dimensions of the TLS ecosystem, improve scanning methodologies, and derive further working challenges. This includes measuring the impact of usability problems and deployed TLS configurations, and improving Hypertext Transfer Protocol Secure (HTTPS) usage with automated rule creation. First, we present a scalable methodology to assess the state of TLS mechanisms in the email ecosystem [A1, A2]. We draw a comprehensive picture of the current state of every email-related TLS configuration for the entire IPv4 range with commodity hardware and open-source software. Among others, we analyze dimensions such as deployment quality, cryptographic primitives in use, and adoption of TLS in general. We argue that while the overall trend points in the right direction, there are still many steps needed towards secure TLS adoption in the email ecosystem.

Following the results of the email ecosystem, we analyze the trade-off between the depth of Internet-wide scans and the speed thereof. To better balance these requirements, we improve cipher suite scan methodologies. We implement different optimized strategies for TLS cipher suite scanning that – compared to the current best practice – perform up to 3.2 times faster and with 94% fewer connections used while being able to do exhaustive scanning for many vulnerabilities at once [A3]. We thoroughly evaluate the algorithms using practical scans and an additional simulation for evaluating current cipher suite practices at scale. With this work, full TLS cipher suite scans are brought to a new level, making them a practical tool for further empiric research. We then use measurement techniques in a user study to comprehend why large-scale measurements show poor quality of TLS deployments [A4]. We apply our measurement methodology to the results of participants to classify the quality of single deployments in addition to the underlying qualitative study. Our results prove and analyze the usability problems of deploying

HTTPS. We further conclude that we need a default encrypted web, shifting HTTP to nearly 100% HTTPS [A5]. To support this shift to 100% HTTPS, we create and validate rules for HTTPS Everywhere using crowdsourcing approaches [A6]. With a publicly reachable platform at tlscompare.org, we validate new and existing rules at a large scale. Over approximately five months, we obtain results for more than 7,500 websites, finding that users tend to disagree even regarding binary decisions like whether two websites are similar over port 80 and 443.

While our results incrementally help increase users' privacy online, we identify metadata leakage as an important area to be considered as well. The **second part of this thesis** focuses on the **transmission of personal data and metadata in the network layer**, especially for ISPs, as they have unlimited access to this data in transit. While ISPs could deploy techniques for economic benefits, traffic classification, and network management, they do not pose a threat to users' privacy. However, the same techniques could also be used for privacy violations. In this part of the thesis, we show two frameworks for monitoring such techniques. The first framework focuses on techniques within a country, focusing on network transparency [A7]. Compared to other projects, it can fully control the client and server endpoint and therefore analyze network behavior in depth. We perform and evaluate multiple measurement periods over the time span of 16 months, including five different Internet products in Austria. We find different questionable methods for some metrics. The second framework focuses on mobile Internet access networks [A9]. Unlike previous work, it geographically decouples the Subscriber Identity Module (SIM) from the cellular modem by redirecting authentication and cryptographic exchanges remotely to a selected SIM. This gives us a scalable, cost-effective, and controlled measurement platform with the ability to test any subscriber with any operator at any location without any moving parts. We showcase the feasibility of such a platform with different use cases leading to surprising results. Our analysis reveals techniques ISPs could use to interfere with metadata and thus the privacy of users.

In the **last part of this thesis**, we focus on **anonymity in transmission and metadata-free communication.** Different technologies for sender anonymity evolved, but only a few influence users' privacy on a large scale. In this thesis, we focus on analyzing the Tor network, as it is the most commonly used tool and well-tested over time. However, as a low-latency anonymity network it is vulnerable to strong passive observers capable of traffic correlation attacks. To refine methods to detect such strong observers, we develop a novel measurement technique that utilizes the RIPE Atlas framework [A8]. We use this network of more than 10,000 probes worldwide to actively perform traceroute commands from and to Tor guard and exit relays to clients and destinations. Based on multiple global scans, our results validate previous results and show the large influence on Tor posed by a limited set of autonomous systems (ASes), that are in a strong position to carry out effective correlation attacks. We provide an additional source of information that can be used together with Border Gateway Protocol (BGP) route information to increase the accuracy of future models and simulations of Tor and ultimately improve anonymity on the Internet.

In conclusion, the **scientific contribution of this thesis** is:

- We refine existing measurement methods for PETs [A3, A8].
- We define, implement, deploy, and evaluate novel measurement frameworks for PETs [A8, A9].
- We gain insights into the causes of poor PETs deployment quality [A4, A5, A6].
- We conduct measurements to indicate the current state of PETs [A1, A2, A6, A7, A8].

1.4.1 List of Publications

Following publications were published in the course of this Ph.D.:

[A1] **Wilfried Mayer**, Aaron Zauner, Martin Schmiedecker, and Markus Huber
No Need for Black Chambers: Testing TLS in the E-mail Ecosystem at Large
in *International Conference on Availability, Reliability and Security (ARES)*, Best Paper Award, 2016 [3],
Lead contribution to the idea; Implemented the scanning framework; Maintained ongoing process of global scanning activity; Result evaluation; Lead author; Paper revision; Presentation.

[A2] **Wilfried Mayer**, Aaron Zauner, Martin Schmiedecker, and Markus Huber
No Need for Black Chambers: Testing TLS in the E-mail Ecosystem at Large
Extended Version, *arXiv preprint*, 2015 [4],
Extended Version: contribution see [A1]

[A3] **Wilfried Mayer** and Martin Schmiedecker
Turning Active TLS Scanning to Eleven
in *International Conference on ICT Systems Security and Privacy Protection (IFIP SEC)*, 2017 [5],
Lead contribution to the idea; Implementation of new scanning techniques; Performed measurements; Evaluation results; Lead author; Paper revision; Presentation.

[A4] Katharina Krombholz, **Wilfried Mayer**, Martin Schmiedecker, and Edgar Weippl
"I Have No Idea What I'm Doing" - On the Usability of Deploying HTTPS
in *USENIX Security Symposium*, 2017 [6],
Recruitement questionaire; Technical study design and setup; Co-supervised lab study; Evaluation and grading of configuration results; Co-authored paper.

[A5] **Wilfried Mayer**, Katharina Krombholz, Martin Schmiedecker, and Edgar Weippl
Securing the Internet, One HTTP 200 OK at a Time
in *USENIX login*, Winter 2017 [7],
Follow up on [A4]; Lead author; Revision;

[**A6**] **Wilfried Mayer** and Martin Schmiedecker
TLScompare: Crowdsourcing Rules for HTTPS Everywhere
in *Workshop on Empirical Research Methods in Information Security (ERMIS)*, 2016 [8],
Lead contribution to the idea; Implemented the crowdsourcing platform; Data evaluation; Lead author; Paper revision; Presentation.

[**A7**] **Wilfried Mayer**, Thomas Schreiber, and Edgar Weippl
A Framework for Monitoring Net Neutrality
in *International Conference on Availability, Reliability and Security (ARES)*, 2018 [9],
Co-designed measurement framework; Measurement hardware setup; Measurement contract management; Long-term maintenance and evaluation; Lead author; Paper revision; Presentation.

[**A8**] **Wilfried Mayer**, Georg Merzdovnik, and Edgar Weippl
Actively Probing Routes for Tor AS-level Adversaries with RIPE Atlas
in *International Conference on ICT Systems Security and Privacy Protection (IFIP SEC)*, 2020 [10],
Lead contribution to the idea; Implementation of measurement techniques; Performed measurements; Evaluation results; Lead author; Paper revision; Presentation.

[**A9**] **Wilfried Mayer**, Adrian Dabrowski, Gabriel Gegenhuber, Edgar Weippl, and Michael Franz
Geographically Decoupled Measurements in Cellular Networks
To be resubmitted at IMC
Contribution to the idea; Contribution to the measurement platform design and implementation; Author.

1.4.2 List of Artefacts

Additionally, the following artefacts were published:

- Source code and data of [A3]
  https://github.com/WilfriedMayer/turning-active-tls-scanning-to-eleven
- Source code of [A6]
  https://github.com/WilfriedMayer/tlscompare
- Source code of [A7]
  https://github.com/sbaresearch/monitoring-net-neutrality
- Source code and data of [A8]
  https://github.com/sbaresearch/ripe-tor

1.5 Structure of this Thesis

The remainder of this thesis is structured as follows:

Chapter 2 presents the background and gives an overview of existing literature for the rest of the thesis. This includes TLS measurements, Net Neutrality, and mobile Internet services, as well as the Tor anonymity network.

Chapter 3 addresses measurement results, measurement improvements, and implications in the TLS ecosystem.

Chapter 4 describes two measurement frameworks to analyze transmissions on the network layer and the privacy implications set by ISPs.

Chapter 5 presents the results of global, AS-based measurements for the Tor anonymity network, possibly affecting the anonymity of users.

Finally, Chapter 6 discusses the different aspects of conducted privacy measurements, gives an outlook on future work, and concludes this thesis.

CHAPTER 2

Background and Literature Review

In this thesis, we analyze ecosystems of different privacy-enhancing technologies affecting the users' privacy at different stages. Van Blarkom et al. [2] define PETs as: "Privacy-Enhancing Technologies is a system of ICT measures protecting informational privacy by eliminating or minimizing personal data thereby preventing unnecessary or unwanted processing of personal data, without the loss of the functionality of the information system." As we will present in this thesis, using a single PET is not sufficient to protect users' privacy. But the ecosystems of different systems are highly diverse, so in this chapter, we present the background and literature review of the different privacy-enhancing technologies addressed in this thesis. We present the background and related work on *Transport Layer Security* in Section 2.1, of *privacy in the network layer* and related topics, e.g., net-neutrality in Section 2.2 and conclude with the background of *metadata-free communication and anonymity* with its most important example Tor in Section 2.3.

2.1 Transport Layer Security

The ecosystem of TLS ensures confidentiality, authenticity, and integrity of data communicated in networks without the loss of any functionality. It is responsible for the majority of encrypted online communication, not only for HTTPS and email protocols but also numerous other applications.

2.1.1 The TLS ecosystem

TLS and related protocols are specified in a variety of Requests for Comments (RFCs). The most important one is RFC8446 [11]. It defines the most modern version of TLS, version 1.3. With TLS 1.2. published in 2008 (RFC5246 [12]), and TLS 1.3. not until ten

years later in 2018, TLS 1.2. was for long the most recent version of the SSL/TLS protocol family. This gap led to some problems of its own. However, TLS does not solely consist of RFCs specifying the network protocol. The ecosystem around it is very important. An in-depth overview of this ecosystem, and more specifically on its usage in HTTPS, is given by Ristic [13] and Clark et al. [14]. The different parts of this ecosystem can be seen in regard to the principle of the weakest link. When one part of the system breaks, TLS cannot fulfill its goals, although other parts work flawlessly. A selection of different viewpoints on the TLS ecosystem are, e.g., (i) the TLS protocol specification itself, (ii) the underlying cryptographic primitives (e.g., RSA, Diffie-Hellman Key Exchange, Advanced Encryption Standard (AES), ...), (iii) the public key infrastructure which facilitates trust, (iv) software security of specific implementations, (v) user behavior while using the protocol, (vi) the security of application layer protocols which use TLS, (vii) and the configuration and infrastructure of the deployed instances of TLS on a global scale.

One of the core functionalities of TLS lies in the TLS handshake, where a TLS connection is built, and cryptographic functions are negotiated. TLS supports extensibility, i.e., the possibility of exchanging the used cryptographic functions. This extensibility is accomplished through the concept of cipher suites. A cipher suite is represented by a two-byte value and determines the cryptographic primitives to be used in the connection [15,16]. It consists of methods for key exchange, encryption, and message authentication. For example, the cipher suite `TLS_ECDHE_RSA_AES_128_GCM_SHA256` (Internet Assigned Numbers Authority (IANA) assigned values: `0xC0, 0x2F`) combines an Elliptic-curve Diffie–Hellman Ephemeral (ECDHE) key exchange with RSA authentication, AES in Galois/Counter mode (GCM) for encryption, and SHA-256 for message authentication (HMAC). Historically, TLS implementations used their own nomenclature for cipher suites as IANA only standardized TLS parameters in 2006 [17] with the introduction of TLS 1.1. The used cipher suite and the TLS version are negotiated in the TLS handshake. First, the client sends a message including a list of supported cipher suites. Second, the server replies with a message choosing one of these cipher suites. These cryptographic primitives are subsequently used. A large number of cipher suites exist, and they can be used in different TLS versions. Some attempts to eliminate old cryptography were made, e.g., the ban of export-grade crypto in modern TLS versions or RFC7465 [18] forbids the use of the insecure RC4 cipher. Nevertheless, the most significant change in the cipher suite concept was introduced with TLS 1.3, leading to a new – and more secure – set of cipher suites that can only be used in TLS 1.3.

2.1.2 Problems within the TLS ecosystem

TLS has a long history of incidents, flaws, and problems. These problems can be attributed to different viewpoints of the TLS ecosystem as described above. However, an overview of certificate trust model problems and enhancements is given by Clark and Oorschot [14] in 2013. A good summarization of attacks on TLS is given by Sheffer et al. [19].

In recent years, the TLS protocol and the underlying crypto have been under scrutiny

by security researchers, and numerous attacks have been discovered: Browser Exploit Against SSL/TLS (BEAST) [20], Padding Oracle On Downgraded Legacy Encryption (POODLE) [21], Compression Ratio Info-leak Made Easy (CRIME) [22], Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext (BREACH) [23], Lucky13 [24], and timing-based side-channel attacks [25]. Associated with the first two viewpoints (i, ii) of the TLS ecosystem described above, some of these exploits make use of the incorrect usage of block cipher modes, compression, and shortcomings in specific versions of TLS. The research area of formal verification is related to these attacks [26,27] leading to formally verified implementations of TLS [28–30]. As the TLS stack is implemented in code, certain TLS implementations are vulnerable to attacks, independently of the protocol specification, see (iv) above. The most famous example of a security-relevant error is Heartbleed [31]. It is a devastating bug in OpenSSL which allows attackers to read content from the server process's heap memory, including login credentials and the private key used by the server. Other less known software attacks exist, e.g., MS14-066 [32] (dubbed "Winshock") which allowed a remote attacker code execution on a system by supplying a malicious Elliptic Curve Digital Signature Algorithm (ECDSA) certificate during the handshake. Other problems of TLS are often caused by the use of old and deprecated features of TLS, e.g., the support of export-grade algorithms, old TLS versions, or insecure ciphers. Two examples are POODLE [21] caused by the use of deprecated SSL 3 and Decrypting RSA with Obsolete and Weakened eNcryption (DROWN) [33] caused by the active support of SSL 2. A secure configuration is non-trivial; therefore, several guidelines give recommendations on how to (i) configure cipher suite settings and (ii) improve the configuration of TLS-enabled server applications [34,35]. However, these methods all rely on the administrator to actively improve the setup by changing the configuration, including the supported cipher suites manually, see (vii) above. Hence, the ecosystem is adapting slowly.

2.1.3 Increased transparency through scanning

Because TLS is used globally and many problems are not visible on a small scale, the TLS ecosystem was utterly intransparent. More transparency was needed, and one way to achieve this is TLS-related Internet-wide scanning endeavors. The first efforts started in 2007 when Lee et al. [36] surveyed 19,000 HTTPS-enabled servers and inspected the supported TLS versions and different cryptographic primitives. With only 19,000 evaluated servers, this is a long way from today. One of the first censuses of edge hosts of the IPv4 address space was given by Heidemann et al. [37] in 2008.

Nevertheless, the size of measurement studies increased constantly, with larger studies conducted by the Electronic Frontier Foundation (EFF) Observatory [38], which collected 1.3 million unique certificates of the HTTPS/TLS public key infrastructure a few years later. In 2011, a first in-depth analysis on the Public Key Infrastructure (PKI) of TLS was given by Holz et al. [39]. Amann et al. [40] revisited SSL and X.509 deployments in 2012 with passively collected data. Also, additional passive data was taken into account (Amann et al. [40,41]).

Starting with 2013, the studies of Durumeric et al. became a gamechanger. With new scanning methods (e.g., *zmap* [42]), studies were suddenly able to cover the IPv4-wide Internet. These methods implemented new ideas, e.g., not using per-connection states, and improved the speed and quality of large-scale scans. Studies that used this new scanning behavior are, e.g., the certificate ecosystem study by Durumeric et al. [43], and studies on vulnerabilities like Heartbleed [31] that solely examine one exclusive issue. Zhang et al. [44] used one of the scan results of the HTTPS ecosystem study to find untrusted certificates. These certificates were used as a metric, among others, to find a relationship between misconfiguration and malicious activity of networks.

Several studies examined additional deployed security mechanisms. Huang et al. [45] surveyed forward secrecy parameters and measured latencies of different cipher suites, Kranch and Bonneau [46] conducted a study of HTTP Strict Transport Security (HSTS) and public key pinning deployments. Weissbacher et al. [47] analyzed the adoption of Content Security Policy (CSP). But these datasets are all limited to the Alexa Top 1 Million ranking and, in the case of CSP and HSTS, not relevant for TLS in general. Heninger et al. [48] performed Internet-wide scans to identify vulnerable RSA and Digital Signature Algorithm (DSA) keys and were able to obtain 0.5% of all RSA private keys for HTTPS. Other papers analyzed, at large, the unsound use of cryptographic primitives like the Diffie-Hellman key exchange parameters [49] or problematic issues with RC4 [50] or RSA in TLS [48, 51, 52]. Adrian et al. [49] investigated the security of Diffie-Hellman key exchanges using an Alexa Top 1 Million scan.

Problems range from using old and deprecated TLS versions [53] to the use of insecure cryptographic primitives like export-grade ciphers [3, 27] or weak RSA keys in certificates [49], to the use of TLS implementations with known bugs like Heartbleed [31].

Other projects like the Qualys SSLTest [54] also scanned full TLS configurations, but these projects are designed for single-host configuration tests and not for Internet-wide studies. In contrast, newer studies tried to draw a complete picture of the certificate ecosystem [55] while missing some underlying security primitives. Durumeric et al. [56] also introduced Censys, a search engine for Internet-wide scans, which developed into a commercialized project. It also featured community-defined, pluggable scanning modules and user-defined annotations for existing scan results.

With the advent of large-scale Internet-wide scanning, also the scanning projects themselves were studied. Dainotti et al. [57] measured the impact of a coordinated Internet-wide scan using the University of California, San Diego (UCSD) network telescope (a "globally routed, but lightly utilized /8 network") receiving unsolicited traffic. Durumeric et al. [58] observed Internet-wide scanning activity, identified patterns, and analyzed defensive behavior. Huang et al. [45] describe the results of a complete cipher suite scan to measure perfect forward-secrecy support but again scanned only hosts from the Alexa top 1 million list [59].

Port scanning is the fundamental step for Internet-wide scanning studies and has a history of its own. In the early days, *nmap* was used to perform these types of scans on a larger

scale, but it is relatively slow and does not scale to a more significant number of hosts in a reasonable time. The breakthrough was achieved with the already mentioned development of *zmap* [42] and *masscan* [60]. Both tools use new methods to optimize large-scale scanning and are so far mainly used for port and vulnerability scanning. However, these improved methods are not directly applicable to fine-grained TLS scanning. With *zgrab*, it is possible to establish TLS connections, but it is still not the most optimal way for examining full cipher suite configurations. Other tools implemented algorithms that conduct a full scan of all cipher suites by using one TLS handshake for each cipher suite, which is slow and produces a lot of traffic, hence there is a lot of potential for optimization.

2.1.4 The problem of not using TLS

Several improvements were introduced to ensure the usage of HTTPS where possible and encourage the use of HTTPS. Server-side solutions are available to prevent cleartext communication despite server-side redirects. For example, HSTS [61] was standardized to enforce the use of HTTPS. Still, it relies on Trust on first use (TOFU).

With HSTS preloading, browsers ship predefined HSTS lists to enforce HTTPS before the first use [62]. In contrast to these server-side solutions, HTTPS Everywhere [63] – developed by the EFF and the Tor project – has to be installed in client browsers. It is a browser extension that automatically connects to the encrypted variant of a website for domains that match a set of user-defined rules. It ensures that a secure connection is used if available by rewriting Uniform Resource Locators (URLs) in the browser before attempting to start a connection over HTTP. These rules are handcrafted because the behavior of interactive websites may change or page loads are not supported. This plugin can be deprecated with the shift of the web to an HTTPS-default one, but it is needed for the transition phase. The increased adoption of HSTS and especially HSTS preload lists with HSTS preloaded top-level domains can be seen as more powerful. Nevertheless, both solutions promote the use of HTTPS.

2.1.5 Non-HTTPS protocols and TLS

While often only associated with HTTPS, TLS is used in different application layer protocols. A particular focus on the usage of TLS on smartphone and non-browser software led to multiple studies [64–67]. Also, the email ecosystem uses TLS for numerous protocols: Simple Mail Transfer Protocol (SMTP) is used for transmitting emails between servers and usually runs over Transmission Control Protocol (TCP) port 25 [68, 69]. Clients use either the Internet Message Access Protocol (IMAP) or the Post Office Protocol 3 (POP3) to retrieve emails from a message delivery server and SMTP for email submission. IMAP uses TCP ports 143 or 993 [70, 71], POP3 uses TCP ports 110 or 995 [71, 72], and SMTP Message Submission runs on TCP ports 25, 587 [73], or 465. Although IANA revoked the usage of port 465 for SMTPS in 1998, it is still used for this purpose. All client-facing protocols are available on two different TCP ports

Port	TLS	Protocol	Usage
25	STARTTLS	SMTP	Email transmission
110	STARTTLS	POP3	Email retrieval
143	STARTTLS	IMAP	Email retrieval
587	STARTTLS	SMTP	Email submission

Table 2.1: Email related ports supporting opportunistic encryption with STARTTLS

because there are two distinct ways to establish a TLS session: implicit TLS, where the connection is established with a TLS handshake, and in-band upgrade of the used plaintext protocol via the STARTTLS command. The flow of STARTTLS messages is specific to the underlying protocol to be secured and not a general way for an in-band upgrade. Different RFCs [69, 71] specify these details. The term "opportunistic encryption" [74] is commonly used when referring to TLS in email protocols. Opportunistic encryption means that connections are usually not authenticated (certificates not validated) and are vulnerable to an active attacker but enable more widespread use and deployment of security protocols like TLS and efficiently secure against passive adversaries. Table 2.1 shows an overview of commonly used ports for email that can be secured with STARTTLS.

In recent years also other Internet protocols were secured by TLS. Two protocols secure the communication for the Domain Name System (DNS), namely DNS over HTTPS (DoH) and DNS over TLS (DoT). These protocols are not to be confused with Domain Name System Security Extensions (DNSSEC), which secures the authenticity and integrity of DNS. DoH and DoT support DNS requests via HTTPS, respectively TLS.

Studies for the use of TLS within email protocols were also conducted. Durumeric et al. [75] examined the support of Sender Policy Framework (SPF), DomainKeys Identified Mail (DKIM), Domain-based Message Authentication, Reporting and Conformance (DMARC), and STARTTLS for email transfer with SMTP. The set of scanned email servers is also built upon the Alexa Top 1 million domains. With an additional data set from Gmail, they also analyzed the use of inbound and outbound TLS traffic on the Gmail servers. This also included the used cipher suites for inbound traffic. They did not cover client-to-server use cases for email submission and retrieval and did not study all accepted cipher suites for all email-related deployments. Concurrently with our work [3], Holz et al. [53] analyzed the use of TLS for communication protocols. Their dataset included an active scan with zmap and OpenSSL as well as passively collected traffic of one university network.

2.2 Privacy in the Network Layer

As the users' communication is transmitted over the Internet, ISPs could read the data and potentially tamper with it. This affects the users' privacy. Questionable techniques

that would allow ISPs to interfere with the users' privacy can also be used for other reasons, e.g., economic advantages of the ISP. Many cases of ISPs using such questionable techniques are known. Often cases of violation are only mentioned in media or on various blogs. These reports give good insights into the practices of Internet service providers. They get short-span attention but don't scientifically examine and quantify the ISP's behavior.

Others are described by the scientific community: In 2010, Dischinger et al. [76] built the tool *Glasnost* to detect ISPs differentiating traffic. With over 350k users, they were able to detect BitTorrent differentiation. In 2011, Weaver et al. [77] studied a practice where ISPs – and several companies enabled by the ISPs – redirect failed DNS lookups for monetization. In 2015, Kakhki et al. [78] studied traffic differentiation in mobile networks. They tested 12 ISPs in 5 countries by replaying prerecorded application data. They found that interference from middleboxes is pervasive. Xu et al. [79] investigated the detailed behavior of web proxies used in cellular networks. This includes parameters like object caching, traffic redirection, image compression, and connection reuse. In 2016, Kakhki et al. [80] specialized in T-Mobile's BingeOn program. This US-based service zero-rated different video streaming services. They summarize the used techniques and introduce the concept of free-riding by exploiting BingeOn's classifications. Nakibly et al. [81] investigated ISPs that injected false content, e.g., advertisements, into the data stream. Although not differentiated by ISPs, Khattak et al. [82] researched the different treatment of Tor users, which is also relevant for this thesis. Also, in 2016, Sahin and Francillon [83] measured so-called *Over-the-top (OTT)* bypass fraud. Telephone calls get fraudulently redirected to OTT services, bypassing terminating operators. In 2019, Li et al. [84] identified the lack of tools to independently audit net neutrality policies. They developed a mobile application called *Wehe* to gather over 1 million crowdsourced measurements. Then, they combined this data and identified traffic differentiation at 30 ISPs. The concept of traffic policing – dropping packets in contrast to traffic shaping – was studied by Flach et al. [85]. Studies related to Internet censorship, e.g., Filastò and Applebaum [86] are relevant for this thesis. They created the Open Observatory for Network Interference (OONI), a censorship analysis tool.

The mentioned studies are often conducted with a narrow focus, using their own measurement platforms. However, some open measurement platforms exist, giving researchers the possibility to conduct measurements openly. We introduce two of the measurement platforms related to this thesis.

The Réseaux IP Européens (RIPE) Network Coordination Centre introduced the RIPE Atlas measurement platform [87, 88] in 2010. They distributed a large number of simple measurement probes allowing a distributed execution of a small set of network commands, e.g., traceroute. Users can create their self-defined measurements that get accounted for with platform-specific credits. It uses open methodologies in a free and open-source software environment. RIPE Atlas continuously improved [89] in the last decade. Ten years after its introduction, it consists out of approx. 11,000 measurement *probes* and 650 *anchors* and is running a large set of measurements leading to a vast amount of

data. It is utilized by a multitude of different research projects, covering a diverse set of research questions. This includes among others: network routing information [90], various protocol measurements [91]. Also, the censorship detection study by Anderson et al. [92] made use of the RIPE Atlas network and measured blocking events in Turkey and Russia. However, RIPE Atlas is focused on landline Internet connections and only allows a small set of possible measurement commands.

In 2015, Alay et al. [93] presented the MONROE measurement platform. Further specifications were published in 2016 [94] specifying the architecture and measurement node design. They present it as an open, European-scale, and flexible measurement platform for mobile broadband networks. However, the architecture of MONROE is limited as each node is physically connected to the measured SIM cards. In 2020, members of the project are still located in only four European countries [95].

So, we identified several factors that could be improved through measurements via fully controllable, large-scale measurement platforms as described in this thesis.

2.3 Anonymity Overlay Networks

A lot of related work for metadata leakage and metadata-free communication (anonymous communication) exists. We will focus on anonymity systems and Tor, including the problems of the used routing algorithms. A breakdown of the different terminology used for anonymity is given by Pfitzmann and Hansen [96].

There have been many proposed solutions to solve the challenges regarding online anonymity and privacy, but most of them fail to address the relevant issues for users. Among its first was the pseudonymous remailer anon.penet.fi [97]. While this anonymity system was a mere proof-of-concept for a pseudonymous remailer, it quickly gained popularity until it was finally shut down in 1996. Since then, numerous systems have been proposed to enhance the privacy of email communication like Mixmaster or Mixminion [98], as well as numerous others [99]. While Mixminion included some specific constructions to enhance privacy at scale and for everyone participating, e.g., forward anonymity, the prevalent usage of TLS on links, or dummy traffic, it never gained widespread usage.

The currently most important low-latency anonymity network – and as such an essential tool for ensuring metadata-free communication – is *Tor*.

Tor, described originally by Dingledine et al. [100] in 2004, grew to the most important anonymity system online nowadays.

Media often shifts the focus to deployed use cases of Tor, namely hidden market places. Researchers found that hidden services like, e.g., the meanwhile discontinued Silk Road, are mainly used for selling light drugs and petty crimes [101–103], while reports in the media are often exaggerated. Most recently, crawling frameworks for hidden services [104] and their privacy implications have been presented, with OnionScan [105] being the most advanced to date.

As a low-latency overlay network, it is inherently vulnerable to passive attacks by global observers, which is already described in the original specification. Therefore, they work with a threat model that includes attackers that can only observe fractions of the network traffic. Feamster and Dingledine [106] provided the first analysis of location diversity in the Tor network for independently operated ASes based on BGP routing tables. They analyzed the probability of an entry path to the network and an exit path from the network will cross through the same AS. Their analysis shows that previous methods of choosing paths/nodes based on IP prefixes are not sufficient to guarantee a diverse set of ASes, since in about 10% to 30% of the time both the entry and exit path to the mix network will cross through the same AS. A refinement of this approach by Edman and Syverson in 2009 [107] shows that the previous study even underestimated the potential threat. A study of Tor security properties against traffic correlation attacks was presented by Johnson et al. [108]. Their results show that, depending on location, a user's chance of compromise can be at 95% within three months of monitoring against a single AS. One mitigation they proposed is to carefully select which entry and exit nodes to use. Wacek et al. [109] built a graph of the Tor network to capture the network's AS boundaries. Using this graph, they provide an evaluation of a set of proposed relay selection methods and quantify their respective anonymity properties. Their results show that bandwidth is an important property for the performance of such algorithms and should not be neglected.

The importance of location diversity in the Tor network has been shown by several attacks proposed in recent years. Vanbever et al. [110] provide a study of the capabilities of AS-level adversaries. Sun et al. [111] describe a set of advanced routing attacks on Tor, named *Raptor*. They also describe the feasibility of asymmetric AS-level attacks by observing not only data traffic from the exit relay to the server but also TCP acknowledgment traffic on other routes, which increases the capabilities of AS-level adversaries. In 2016, Nithyanand et al. [112] also used data on the Internet's topology [113] in combination with AS-topology simulations [114] to estimate the threat posed by adversaries to Tor users. While previous attempts at the correlation of traffic [115, 116] had very limited performance or required a large amount of captured traffic or time, *DeepCorr* [117], developed by Nasr et al., greatly improves the feasibility of such attacks. By leveraging emerging learning mechanisms, they manage to achieve drastically higher performance compared to existing state-of-the-art systems.

To mitigate the threat posed by an AS to be able to monitor Tor users, various kinds of protection mechanisms have been proposed [118]. Nithyanand et al. proposed *Astoria* [112], an AS-aware Tor client. While similar in functionality to *LASTor* [119], it provides improved protection with concern to threat models and attacker capabilities. Sun et al. [120] presented a measurement study on the security of Tor against BGP hijacking attacks and presented a new relay selection mechanism to mitigate such attacks on Tor. In contrast to previous approaches, *DeNASA* from Barton et al. [121] provides a mechanism for AS-aware path selection independently of the destination. Additionally, they proposed another system for the creation of efficient and anonymous Tor circuits [122].

Also, Johnson et al. [123] worked on improving path selection algorithms. Annessi and Schmiedecker [124] also tried to increase the performance. Also, the design of Tor has been theoretically improved and redesigned for improved performance demands [125]. Hanley et al. [126] proposed an extension to the work presented by Sun et al. [120] to increase the provided privacy and anonymity guarantees. Wan et al. [127] showed that several attacks against a set of the proposed protections were still possible, but they also proposed simple solutions, which allow mitigating the threat posed by their developed methods.

The combination of multiple privacy-enhancing technologies is also recognized by research. Winter et al. [128] studied the impact of malicious Tor exit nodes and confirmed the importance of an encrypted last mile with, e.g., HTTPS. Also, Huber et al. [129] showed the importance of HTTPS by showing the deanonymization of Tor users who browsed the web via HTTP.

CHAPTER 3

Analyzing Transport Layer Security

Transport Layer Security is the foundation of privacy protection in today's web and is protecting the privacy of all Internet users. The main application – and highly researched – is the use within the application layer protocol HTTPS. But TLS is independent of the application layer protocol and is used by many other protocols. We identify another large deployment area in the e-mail ecosystem, which didn't get as much attention from the research community.

To address this research gap, we first describe the results of an e-mail ecosystem measurement study in Section 3.1. The results are meaningful, but longitudinal data is missing. Large-scale cipher suite scanning did not scale, and therefore, we identify potential improvements in large-scale TLS cipher suite scanning methodology.

We show such improvements in Section 3.2 and evaluate performance gains compared to other studies. This improvement supports reoccurring measurements and consequently supports a better understanding of the essential privacy-enhancing technology TLS. With the results of our studies, we also identify problems in deployed TLS configurations.

In Section 3.3, we discuss findings from our usability studies, in which we evaluate TLS configurations created by participants by cipher suite usage, among others. We also give an outlook on the future of TLS and emphasize the importance of an all-encrypted web.

Last, we look at ways for individuals and web users to improve their HTTPS usage on the client side. Therefore, in Section 3.4, we introduce an approach to generate HTTPS rewrite rules with crowdsourcing mechanisms if automated mechanisms do not work.

3.1 Testing TLS in the E-mail Ecosystem at Large

This section is based on [3, 4] and a follow up on [130].

E-mail has become one of the most fundamental and important services of the Internet and is used by more than a billion users every day. While the implementations and the underlying protocols have (for the most part) endured the test of time, there is no way for users to assess the security and confidentiality of e-mails in transit. Even if the client-to-server connection is secured, server-to-server communication relies on plaintext communication as a fallback, leaving a large fraction of transmitted e-mails unencrypted and not authenticated. This is a serious limitation, as they are passively observable along the transmission path and users are left in the dark whether their e-mails will be transmitted between servers in plaintext or not. While the advent of opportunistic encryption is good from the privacy viewpoint, it offers no protection against active attackers. While Pretty Good Privacy (PGP) would protect e-mails from end to end, its adoption is marginal. Numerous attackers, ranging from local attackers who are able to sniff or modify WiFi traffic, to customer as well as transit ISPs up to nation states are thus able to not only read e-mails in massive quantities, but also to modify, delay or delete them at will. If somehow the usage of TLS between all links during e-mail transit could be enforced, this would not only provide authentication but also confidentiality thanks to the use of encryption. However, due to the decentralized nature of e-mail there is no way to upgrade all servers or somehow enforce link encryption. In fact, we are still just beginning to understand the consequences of large fractions of e-mails being transmitted unencrypted.

In this section, we summarize and extend the results of our study of cryptographic primitives for e-mail at scale. While many of the problems of e-mail are connected to the design of popular e-mail protocols, we are the first to quantify and evaluate the adoption of well-known security primitives on a large scale, namely the support of TLS for e-mail protocols on the Internet. Previous large-scale studies on usage and prevalence of TLS focused either on the protocol itself or its usage and implications for secure web browsing using HTTPS. W present our findings on TLS usage and deployment in e-mail protocols using active probing. In particular, the contributions of this section are as follows:

- We summarize the results of our large-scale study of TLS usage in the e-mail ecosystem.
- We model the graph of all TLS-enabled SMTP servers and find that it is much more complicated to eradicate plaintext and insecure cipher suites then with other protocols.
- We evaluate openly available Internet-wide certificate information for TLS in e-mail over a 6-month period.

The remainder of this section is organized as follows:

In Section 3.1.1 we summarize the scanning methodology and the different inputs we used. Section 3.1.2 recapitulates our results of multiple months of scanning activity and extends the conclusions we make upon our results. Section 3.1.3 discusses our findings, interprets the results and compares them with other protocols that use TLS. Section 3.1.6 introduces mitigation strategies and methods how the overall security for e-mail can be increased.

3.1.1 Methodology

Here, we describe the source of our input data, the methodology and tools we used, and our considerations regarding the impact of our scans. In the beginning we started with the following requirements: the performance of our methodology should be sufficient in that it runs on commodity hardware with a fast Internet connection. Our goal was to scan more than a million hosts in less than a week, with low additional load for the target systems. Handling network connections is what these systems do (by design), and we took specific precautions that our connections would have no negative impact on the performance or operations. Furthermore our scans should be complete, as in all cipher suites should be covered as well as all the important TLS parameters should be collected i.e., certificates, the certificate chain up to the root Certificate Authority (CA) which signed the certificate, RSA primes and DH-group. For our evaluation we wanted to cover all seven ports which are used by default for e-mail.

Input data-set

The source of our scan consists of a list of all input IP addresses together with the application-specific TCP port. Of course not all IPv4 addresses support an application or even accept incoming data on an application-specific port. We therefore would have to conduct a regular port scan. With the introduction of zmap [42] and similar tools, IPv4-wide port scans can be accomplished quite fast. Many results of port scans are publicly available on a short-term basis, so we decided to use existing datasets from scans.io[1] as a source for our cipher suite scans. The "Full IPv4 Banner Grab and Start-TLS" and the "Full IPv4 TLS Handshake and Banner Grab" datasets from the University of Michigan and the Project Sonar Study (Port 465) from Rapid7 Labs consist of all addressable IPv4 hosts which completed a full TLS handshake at a given time. We used results originating from March to April 2015 as the initial set of IPs for our scans. To further tighten the set of input addresses, we only selected hosts which completed the TLS handshake without any error by filtering the included error key. This resulted in 18,430,143 IP/port combinations which were used for our scan.

[1] https://scans.io

In June 2015 we verified these IP/port combinations with the results of banner grabbing and certificate collection scans performed with masscan[2] by our own team. This tool uses asynchronous connections, its own custom user-land TCP/IP stack and other approaches to speed up IP address and port enumeration. It can also establish a TCP connection to a target system afterwards for e.g., banner collection. This scan resulted in 23,367,809 possible IP/port combinations. Nearly 70% of these combinations were already included in our initial input data. Because no existing scan for SMTP Port 587 (client-facing message submission) was publicly available, we used the results of a masscan run for input to an additional cipher suite scan on port 587 in June 2015. To allow observations based on the domain of a host, we also scanned mail exchanger (MX) records for all domains listed in the Alexa Top 1 Million ranking. We stored these records together with the resolved IP addresses to allow the interpretation of our scan result with the popularity of a domain, and to combine them with the Google Transparency Report on safer e-mail [131].

Scanning the Cipher Suite Support

The specification of the TLS handshake protocol [12] requires that the client specifies a set of supported cipher suites in the `ClientHello` handshake message and the server then picks a specific one of them. So the client has to initiate more than one handshake to determine which set of cipher suites for a specific TLS version a server exactly accepts or rejects. The simplest algorithm is to test each cipher suite individually by offering exactly one cipher suite per `ClientHello`. The response is either a `ServerHello` or `Alert` message which can be interpreted as an accepted or rejected cipher suite. We used the tool *sslyze*[3] to perform this type of scan. It is a plugin based tool written in Python, built around a custom OpenSSL wrapper. The plugin "PluginOpenSSLCipherSuites" was used to test a specific set of cipher suites. Seven different cipher suites were tested for SSL 2 and 136 cipher suites for SSL 3, TLS 1, TLS 1.1 and TLS 1.2 each. This deliberately also includes combinations of TLS version and cipher suite which are not specified in the corresponding RFCs and thus violate the standard e.g., SSL 3 with ECDHE cipher suites. In total we established at least 551 connections per IP to fully test the supported cipher suites for each IP/port combination. We stored the acceptance status as either accepted, rejected, preferred or error. If the handshake was successful we also stored key exchange parameters for the Diffie-Hellman key exchanges. In the case that the TLS connections were not successful, we stored the alert message or error reason given. Additionally we stored the complete certificate chain per IP.

To optimize *sslyze* for our needs we made certain changes. *sslyze* was written as a command line tool. To use it efficiently with millions of possible targets we embedded the tool into a distributed environment and integrated it with a queuing system. As such our methodology can be easily used in a distributed environment, with many simultaneous

[2] `https://github.com/robertdavidgraham/masscan`
[3] `https://github.com/nabla-c0d3/sslyze`

hosts that perform scanning activity working together on an input list of IPs to be scanned. We deliberately did not try to hide our scans, and decided to scan the hosts from one host only. We only used one IP for the scans (Ubuntu Linux 14.04, Intel Core i7, 4x 3.10 GHz, 32GB DDR3 RAM, 100MBit/s Network Bandwidth). We consider our results to be a lower bound for assessing the usage of TLS in e-mail, and the scan time can easily be decreased by adding more hosts or bandwidth. Other hosts were used for postprocessing.

sslyze scans all cipher suites on a per host/port basis, and thus fullfils our requirements listed above. All cipher suite tests are queued consecutively within one process. It then maximizes parallelization by using 15 threads per process and by using more than one process. This intense scan behavior may be seen as a Denial-of-Service attack and may disrupt normal service behavior because all connection attempts are started in a very short timeframe. To solve this issue, we had to spread the cipher suite tests over time. Also, the availability of hosts may change during a scan, which can increase the chance of incomplete results due to hosts that go offline during that timeframe. We therefore serialized the behavior of the scan, spreading the total scan time to several minutes for one host/port, and increased the total number of scan processes. This yielded a good balance between minimizing the scanning impact on the target systems to be scanned – as not all connections are opened in direct succession – while completing the entire list of cipher suites to be scanned in reasonable time. With this methodology we were able to scan between 12,000 and 30,000 IP/port combinations per hour. Our bottleneck was identified to be the number of sockets provided by the operating system, in particular the time until a socket can be reused - spreading the load over multiple machines can easily help to overcome this obstacle. After scanning was completed for each port we transformed the output and analyzed it using Google BigQuery[4].

Ethical Considerations

Similar to related work on active probing on the Internet, we took specific measures to inform not only about our intentions, but also about our scanning methodology and how people can be excluded from future scans. On our scanning host we served a simple webpage to inform about our scans, both over HTTP and HTTPS. We furthermore created a specific abuse address as single point of contact and included it in the WHOIS abuse database. Lastly we set a reverse DNS entry to identify us as a scanning host and to point to the webpage. We were in direct contact with our upstream ISP's network administrators to answer complaints and inquiries and made sure that all requests were answered in a timely manner. To prevent connection- or resource-wise Denial of Service for the hosts to be scanned, we tested our scanning methodology on devices with constrained hardware, in addition to spreading the number of connections over time. These devices were limited in the overall computation resources. We chose them to reflect, appropriately, the smallest possible devices on the Internet that could be used to run a

[4] https://cloud.google.com/bigquery/

full e-mail stack, similar to a very small cloud instance or an embedded system (router or appliances). We did not take any specific measures to prevent our connections from being logged, and did not take any measures to stay below the radar of vigilant system administrators.

3.1.2 Evaluation

This section presents the data we obtained through our scans. First we summarize the information on the datasets and the direct measurement results. After that, we extend the results by analyzing problems of TLS usage in related e-mail protocols.

In total we conducted over 20 million scans for seven different TCP ports, performed between April and August 2015. Thereof, 18,381,936 were valid results, meaning that at least one TLS handshake was completed. 1,888,832 of the scans were invalid with no TLS session established at all.

In former work [130], we already described detailed results for the following categories:

1. Supported cipher suites
2. Cryptographic primitives
3. Key Exchange Parameters
4. Weak Diffie-Hellman parameters
5. Elliptic Curve Diffie-Hellman
6. TLS Certificates
7. Weak RSA Keys
8. Domains contained in certificates

Certificate volatility

We extend the work by first looking at certificate volatility. To analyze the change in certificate strength over time we used the freely available datasets from scans.io. They cover the last six months on bi-weekly granularity and are based on scans of the entire IPv4 range. We extracted the size of the RSA key of the leaf certificates and plotted them over time for all important e-mail ports. Figure 3.2 shows the key distribution of each port for STARTTLS-enabled client-to-server protocols. While the majority uses 2048 bit strength leaf certificates, only a small fraction uses the stronger 4096 bit size. They seem to be rather stable over time, with a slight increase in the overall number of 2048 bit keys. The key distribution for SMTP is shown in Figure 3.1.

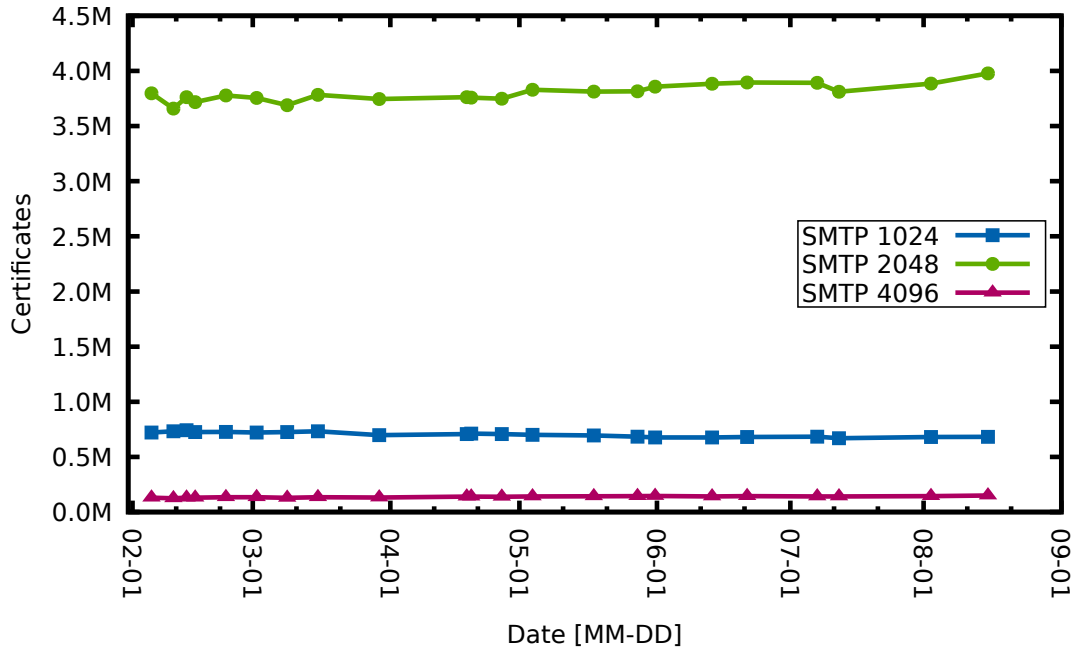


Figure 3.1: SMTP leaf certificate volatility (2015)

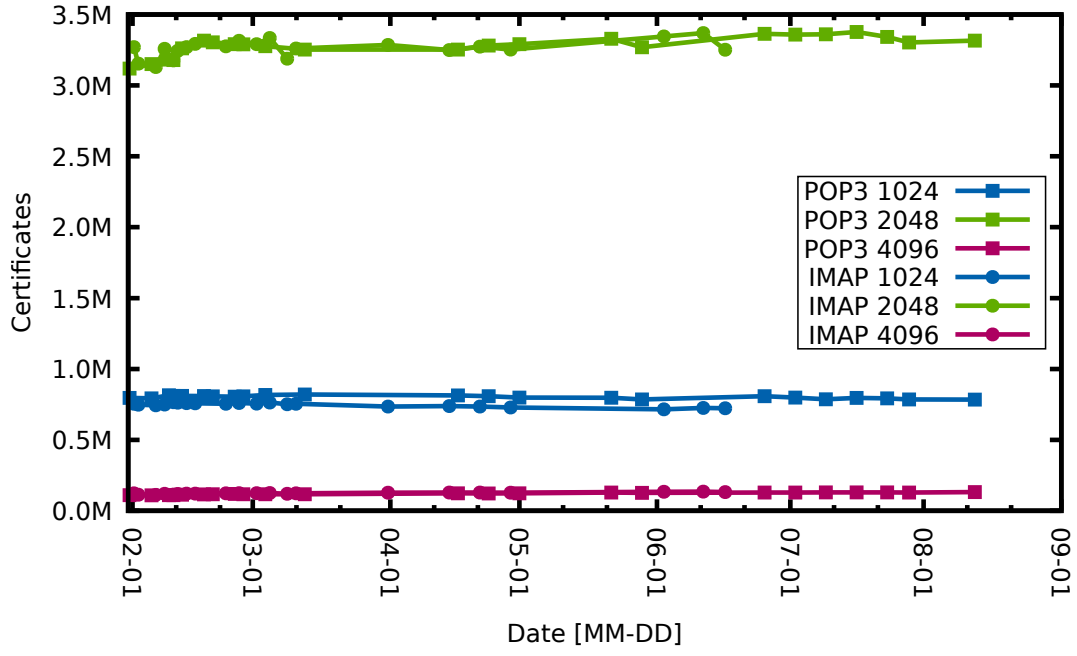


Figure 3.2: IMAP and POP3 leaf certificate volatility (2015)

Plaintext authentication

We further extend our work by looking at plaintext authentication. A commonly observed issue with e-mail are servers supporting plaintext authentication without support for STARTTLS. Despite having nothing to do with cryptography per-se, this is an important issue for transport security as connection authentication data sent via plaintext channel provides an easy target for any attacker. Further, `AUTH-PLAIN` should only be offered after a connection has been upgraded via STARTTLS. If a server offers plaintext authentication on a plaintext connection, neither server nor user will notice if STARTTLS stripping is performed and password data sent in clear over the wire. Table 3.1 outlines the number of hosts that offer `AUTH-PLAIN` and support STARTTLS but allow plaintext authentication before an upgrade to TLS and hosts that do not offer STARTTLS at all.

Port	no STARTTLS	STARTTLS	Total Hosts
25	12.90%	24.21%	7,114,171
110	4.24%	63.86%	5,310,730
143	4.38%	66.97%	4,843,513
587	15.41%	42.80%	2,631,662

Table 3.1: Hosts that offer `AUTH PLAIN`

Cipher Islands

We last extend the work by analyzing cipher suite compatibilities and infer the existence of cipher islands. We define a cipher island as a subset of all SMTP servers that enable a common, specific combination of cipher suites. By deprecating older cipher suites and TLS versions on a single SMTP server, this server cannot negotiate a cipher suite with certain subsets, i.e., cipher islands, and thus uses plaintext transmission. To simulate the consequences of changes in the configured cipher suites between all SMTP servers we build a model to estimate given probabilities based on the data we collected. This is done to calculate the impact in percent that two random servers are able to communicate using TLS if certain cipher suites are allowed or prohibited. We assume that it is equally likely that a random SMTP relay has the objective to transmit an e-mail for any of the others for the sake of brevity (which is clearly not the case). Since these connections can go unrestrictively in both directions – i.e., server X can send a message to any server Y and vice versa – our model still gives useful insights into the consequences of TLS cipher hardening and the estimated use of TLS for specific scenarios as outlined below. To calculate the probabilities we build a weighted graph where the nodes are representing the unique combinations of cipher suites, in total roughly 90,000. The nodes are weighted with the number of SMTP servers that have that specific cipher suite configured. Edges are assigned between two nodes if they share at least one common cipher suite. Table 3.2 shows the initial table from the entire set of SMTP servers on port 25 we collected. The list is sorted by percentage and shows

TLS version	Probability
SSL 3, TLS 1	41.63%
SSL 3, TLS 1, TLS 1.1, TLS 1.2	16.10%
TLS 1	14.75%
Plaintext	10.88%
TLS 1, TLS 1.1, TLS 1.2	8.82%
SSL 3	3.42%
TLS 1.2	2.07%

Table 3.2: Cipher suites in the SMTP dataset

Cipher suite	Probability
Plaintext	52.80%
TLS_ECDHE_RSA_AES256_SHA384	18.47%
TLS_DHE_RSA_AES256_SHA256	11.61%
TLS_ECDHE_RSA_AES256_GCM_SHA384	10.59%
TLS_DHE_RSA_AES256_GCM_SHA384	6.53%

Table 3.3: Cipher suites with SMTP compared to bettercrypto

that the likelihood for any two servers chosen at random to use a cipher suite in TLS version SSL 3 or TLS 1 is 41%. In one out of ten cases the servers are unable to agree on a mutual cipher suite, and e-mail will be transmitted in plaintext. Please note that the actual calculations are conducted on each supported cipher suite, and probabilities are only presented in the aggregated TLS version. Probabilities of less than 1% are not shown.

Simply changing the cipher suites of a specific server can render large percentages of other SMTP servers unreachable, as they can no longer negotiate a bilateral usable TLS cipher suite. Since the fallback in SMTP is to transmit the e-mail in plaintext if there is no agreement on a shared cipher suite, this is clearly undesirable. Assuming that a considerate administrator changes the cipher suites of her e-mail server according to the recommendations from either *bettercrypto.org* [34] (all A-ranked cipher suites) or RFC7525 [35] also the shared cipher suites change. Since the results for both recommendations are extremely similar, the consequences can be seen in Table 3.3. It can be seen that more than half of all SMTP servers that support TLS become unreachable as a consequence.

With the ban of RC4 [18] for use in transport security, we also simulated what happens if RC4 is no longer used in any TLS version. The results can be seen in Table 3.4. As it can be seen, the probabilities do not drastically change compared to the initial Table 3.2. About one in ten e-mails would still be transmitted in plain, but the other supported

TLS version	Probability
SSL 3, TLS 1	40.56%
SSL 3, TLS 1, TLS 1.1, TLS 1.2	16.31%
TLS 1	14.41%
Plaintext	11.44%
TLS 1, TLS 1.1, TLS 1.2	9.04%
SSL 3	2.99%
TLS 1.2	2.41%
SSL 3, TLS 1.1, TLS 1.2	1.10%

Table 3.4: Cipher suites without RC4 in SMTP

cipher suites substitute the lack of RC4 quite well.

3.1.3 Discussion

The results from our scans cast a poor light on the use of TLS in e-mail. Not only are weak encryption mechanism and cipher suites like export grade ciphers supported by a non-negligible fraction of e-mail servers, we also found that the recent increase in HTTPS certificate security (moving certificates from 1024 to 2048 bit) went totally unnoticed for all e-mail related ports, IPv4-wide. Furthermore, millions of hosts are currently misconfigured to allow `AUTH-PLAIN` over unencrypted connections resulting in the risk of transmitting user credentials unencrypted. Even if each of these systems has only one user, this leaves close to 20 million users vulnerable to attack.

Using our model we showed that increasing the cipher suites to the current acceptable level of security would prevent communicating with 50% of all e-mail servers that support and use TLS, since they are no longer reachable over bilaterally supported TLS. This means that the movement to more secure configurations needs to be done on a larger scale, protecting one server alone is insufficient. Administrators should be encouraged to check with which e-mail servers their systems interact regularly, and adapt their security accordingly. We also showed that the entire e-mail ecosystem as a whole is not dependent on RC4 nor SSL 2 nor SSL 3 alone: disabling them comes with minimal impact, and should be considered for the near-term future.

We found that more than 650,000 SMTP deployments accept export grade ciphers with TLS 1.1 and TLS 1.2 (with at least OpenSSL is affected, implementation-wise) although this behaviour is explicitly forbidden (`MUST NOT`) in RFC4346 [132].

We also found troublesome issues despite the cryptographic primitives used (if any at all).

More than half of all certificates we observed were self-signed, meaning that an active attacker could exchange the certificate in a man-in-the-middle scenario [133]. Since the user interface of most e-mail clients lack a convenient way to check the authenticity of certificates, this leaves clients without the most simple protection mechansims – which are well-deployed and well understood for the HTTPS Public Key Infrastructure Exchange (PKIX) ecosystem.

TLS 1.3 was published in 2018, 3 years after our initial scans. Therefore, our data and our suggestions have to be adopted for the current state of transport security in the e-mail ecosystem.

Compared to previous work on HTTPS [38, 43], we did not analyze the CAs which issued the certificates at length. First, we found that a very large percentage of certificates in e-mail protocols are self-signed. Second, while HTTPS enables the client to authenticate the final server serving the content, this is different for e-mail due to the distributed nature of e-mail transmission protocols. Only the first hop is observable to the client, and thus only a fraction of servers along the path of an e-mail. Last, we find this to be rather a policy issue, i.e., who is trusted by the software vendors as certificate authority, than a trust issue, i.e., do we trust CA X to issue certificates for a given domain Y [134]. Furthermore, there are now effective countermeasures available for HTTPS which are transparent for users like HTTP Public Key Pinning (HPKP) [135] for dynamic public key pinning or certificate pinning within the browser to effectively prevent man-in-the-middle attacks. No such mechanisms exist so far for e-mail protocols and server implementations, and as such we purposely didn't evaluate the issuing CAs.

3.1.4 Who leads the way?

The overall security of the e-mail ecosystem cannot be improved by a single actor. Overall security in e-mail transmission can only improve slowly if the majority of transmitting hosts aligns. We identified multiple issues that need to change not only in the long-term, but already in the near-term future. We found that big providers of e-mail infrastructure often lead the way regarding TLS in e-mail. First, the usage of TLS in e-mail has increased since the documents by Edward Snowden on NSA dragnet surveillance were published, with Yahoo and Microsoft *live.com* enabling their server infrastructure to enforce TLS wherever possible. Second, big service providers like Gmail are in the position to give deep insight into the server-to-server use of TLS. With the release of more and more transparency reports (most recently by Twitter), we can draw a more complete picture of TLS use and identify global actors that handle large volume of e-mail and do not support TLS. Based on this data it would be great to have a list of the most important domains regarding legit e-mail volume, similar to the often-cited Alexa list of popular websites.

There will always be space for niche providers like hushmail, riseup.net, or the German Posteo that focus their business model around security and privacy. They target specific privacy-aware users and implement additional security features like DNSSEC, DNS-based Authentication of Named Entities (DANE), DKIM, SPF and more. However, with free e-mail giants like live.com, Google, Yahoo and others that have a business model around targeted advertisement depending on the content of their customers e-mails, they still have an implicit responsibility to protect their customers at least by using and enforcing TLS for e-mail wherever possible (and increasingly do so).

3.1.5 Limitations

Since there is no comprehensive list and global ranking of e-mail domains regarding their e-mail volume and overall popularity, this is a clear drawback of our approach. We treated and weighted all scanned IPs equally. Major e-mail provider that handle an overproportional volume of clients are treated equally to the single-user, single-domain self-hosted server. However, we tried to counter this form of bias in our observations and findings by using for one the most popular e-mail related domains as published by Google, as well as correlate it with the most popular websites, even though there seems to be no direct correlation between popular websites and popular e-mail domains. A full list of popular SMTP hosts would have been very helpful, but without any data openly available we had to resort to unweighted statistical analysis. Another limitation is that our findings are only showing a point-in-time snapshot and as such are only of short-living value.

3.1.6 Mitigation Strategies

Various approaches are technically feasible to increase the security of TLS in e-mail, apart from trying to increase the TLS configuration of each and every individual server. However, in our opinion, the most important issue is to increase the awareness for security and secure configurations among administrators. Furthermore, mitigation can be put into two different categories.

The first class includes various forms of pinning. They could be used (either on the host itself or on the network layer) as the certificate information is transmitted prior to the encrypted communication and the number of e-mails transmitted to particular servers is usually much larger compared to the number of its certificate changes. *Tack.io*[5] is a draft from 2013 which could be used to add trust assertions to certificates and would be especially valuable to protocols making use of STARTTLS.

The second class of defenses includes all methods where signed certificates are transmitted over additional communication channels, most notably DNSSEC or DANE [136]. Both rely on including signed assurances in DNS name records that can be verified by the client to not have been manipulated on the path of communication. Notary services like

[5] `https://tack.io`

International Computer Science Institute (ICSI) Certificate Notary [137], Convergence[6] or the SSL Observatory from the EFF are just a few examples of readily-available and deployed approaches where the trust is distributed among independent entities. Defending against active attackers is much harder however, as the attacker is always in the position to suppress communication like the STARTTLS command at the beginning of the transmission. Such stripping attacks can only be prevented if there is no fallback mechanism to plaintext.

With RFC8314 [138], published in 2018, new standards for e-mail access and e-mail submission are recommended. This includes the deprecation of plaintext protocols and recommends the use of implicit ports. For Mail Transfer Agent (MTA) communication, RFC8461 [139] proposes SMTP MTA Strict Transport Security (MTA-STS), which works similarly to HSTS. MTA-STS enables e-mail providers to declare the ability to receive TLS encrypted e-mails and tells sending e-mail servers whether to refuse unencrypted hosts. Ultimately, with the deprecation of TLS 1.0 and TLS 1.1 in March 2021 by RFC8996 [140], RFC8997 [141] updates the minimum TLS version for e-mail access and e-mail submission to TLS 1.2.

Lastly, operators of smaller e-mail servers often do not change default configurations. They should be incentivized to enable important security primitives, or enforcing TLS for communication. Guidelines to improve security for various software products exist [34, 35, 142], but these suggestions are not automatically implemented. If software products embed secure configurations by default, this could change the overall primitives in TLS usage. Smart update mechanisms could enforce, or at least offer, the use of stronger cryptographic primitives. A promising project for exterminating the usage of self-signed certificates is *Let's encrypt*[7] which is a fully automatic CA to issue trusted certificates for free.

[6] https://convergence.io
[7] https://letsencrypt.org/

3.2 Turning Active TLS Scanning to Eleven

This section is an extended version of [5].

Successful attacks against TLS are irritating the security community on a regular basis. Many of these attacks exploit vulnerabilities in the underlying cryptographic primitives, which, when grouped together, form so-called cipher suites. Often the mitigation of these vulnerarbilites is achieved by simply discontinuing the use of insecure cipher suites. Although easily done, this is a *manual* configuration step, which results in a slowly adopting TLS ecosystem. This progress is only observable through Internet-wide measurements.

Full cipher suite scans are important in order to understand in-depth the TLS ecosystem and the impact of discovered vulnerabilities, as demonstrated in Section 3.1 and by Aviram et al. [33]. Only with detailed information is it possible to thoroughly assess the state of online security, ranging from the security of a single host up to the security of the whole ecosystem. With the recent advent of fast-paced scanning tools it has become possible to proactively scan the entire range of IPv4 on a regular basis. This data is invaluable when reacting to newly released attacks.

In this section, we develop three new scanning algorithms that efficiently test TLS configurations in detail. These full cipher suite scans can then be used to cluster configurations based on different cipher suites, identifying common misconfigurations and facilitate TLS stack fingerprinting. We evaluated these algorithms and estimated the performance gain for Internet-wide full cipher suites scans. We then use these algorithms to scan parts of the IPv4-wide Internet and analyze the results. The specific contributions of this section are:

- We introduce highly optimized scanning methodologies to perform TLS scanning at scale.
- We evaluate our improved methodologies against the top-10k websites and are on average 3.2 times faster than previous methods.
- We show that current cipher suite recommendations are hardly used.
- We publicly release the source code and collected data from our experiments under an open source license[8].

The remainder of this section is organized as follows: Section 3.2.1 introduces our optimized scanning methodologies and the data inputs used for our evaluation. Section 3.2.2 illustrates the achievable gain in overall performance and provides insights into the current TLS deployment. We discuss the results in Section 3.2.3.

[8]The patterns, the mappings and the source code are available online at:
`https://github.com/WilfriedMayer/turning-active-tls-scanning-to-eleven`

3.2.1 Methodology

To improve the scan rate for TLS-specific scanning, we defined the following requirements: First, *time* as the overall time consumption of the scanning process; second, the support for *parallelization* – can different scans be executed in parallel or do they rely on partial results and therefore require a sequential execution? These two requirements are especially important for large-scale scans. Third, the number of *connections* necessary for a scan: How many connections are necessary for a full configuration scan? Also, the generated traffic is derived from the number of connections. Last, the *completeness* of the scan, or how much information we can gather from the results: Is it possible to draw a complete picture of the TLS configuration or is it just one specific detail?

The anticipated use cases range from an interested system administrator or CISO who wants to scan infrastructure for security vulnerabilities up to Internet-wide scans for either specific questions or complete ecosystem analysis.

We identified three existing approaches to scan and identify TLS configurations: The *naive approach* establishes one connection for each cipher suite, starting at the same time. For each connection the server replies with either this cipher suite or with an alert that the cipher suite is not supported. This method is currently implemented by the command line tool *SSLyze* [143]. It highly parallelizes all requests, which results in a fast execution time, especially for non-delaying networks. The number of connections and produced traffic is rather large. This can lead to errors for some hosts, because the number of parallel connections may exceed their limit. This disadvantage leads to error-prone results and affects the completeness in a negative way. *SSLyze* (version 0.12) produces exactly 543 connection attempts to test all cipher suite/TLS version combinations and approx. 500KB of traffic (inbound and outbound) per tested host. These numbers clearly don't scale for Internet-wide scans, making *SSLyze* impractical for this task.

The second approach is implemented by *zmap*. This command line tool, created by Durumeric et al. [42], is primarily used for Internet-wide port scans. With *zgrab* they also implemented an application layer scanner capable of scanning TLS configurations. To minimize the number of connections to exactly one per server, the cipher suites in the `client_hello` message are fixed to a specific research question, e.g., in order to test if RC4 is supported, all cipher suites that use RC4 are included. The server then responds with a `server_hello` message (showing the support of RC4) or an alert (showing that RC4 is not supported). This one-connection-based approach minimizes traffic and performs fast. The downside is that it does not completely evaluate all cipher suites. It is limited in its expressiveness, since only one question per scan can be evaluated.

The third approach is used by the *SSL Server Test* [54]. This web service is designed to test and evaluate one specific web server configuration. Therefore it utilizes the cipher suite settings from different browsers and browser versions as well as settings to test common misconfigurations. It then establishes one TLS connection per setting to

completely evaluate one server configuration. It also includes HTTPS-specific settings and security features, e.g., HSTS or HPKP. The information collected is comprehensive, but the service's design is not suitable for large-scale ecosystem studies.

Introducing New Approaches

We propose the following new approaches for cipher suite scanning:

Connection-Optimal Approach This approach tests all cipher suites per TLS version in a serialized way. The process is illustrated in Figure 3.3. It starts with one TLS `client_hello` message that includes all available cipher suites for this TLS version. The server then responds with one cipher suite that it accepts. The next handshake includes all cipher suites except the one that was accepted earlier. This procedure is repeated until the server does not accept any of the offered cipher suites and responds with an alert. All remaining cipher suites are then evaluated as rejected. This approach uses the optimal, lowest number of connections necessary, but is not parallelizable for one host. Therefore it needs more time, especially for networks with a delayed round trip.

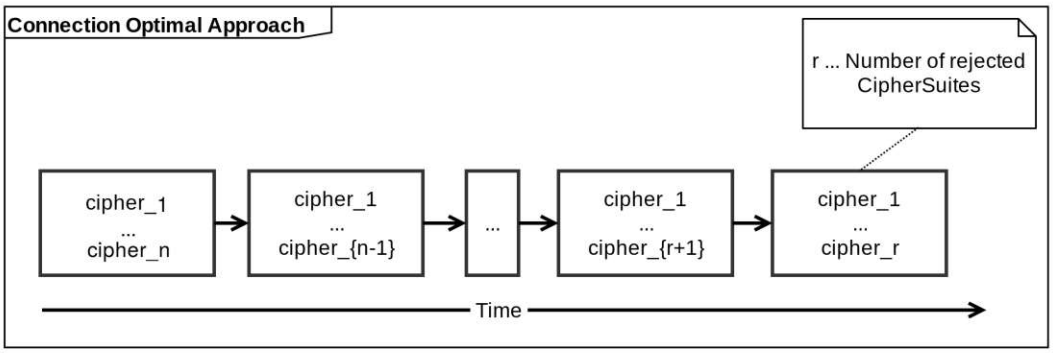


Figure 3.3: Connection-Optimal Approach

Grouped by Cryptographic Primitives The second approach, presented in Figure 3.4, groups cipher suites according to their used cryptographic primitives. It is based on the assumption that server operators disable or enable all cipher suites with a common primitive (e.g., deactivate all RC4-based cipher suites). After the cipher suites are split up in groups, the process follows the methodology of the connection-optimal approach. We currently use groups based on keywords in the cipher suite name, i.e., `SRP`, `PSK`, `EXP`, `NULL`, `(DSA, DSS)`, `(ADH, AECDH)`, `(CAMELLIA, SEED, IDEA, DES-CBC-)`, `RC4`. Primitives that are not supported can be filtered out in the very first round. This approach supports parallel execution of the group tests so that it works with fewer round trips than the connection-optimal approach.

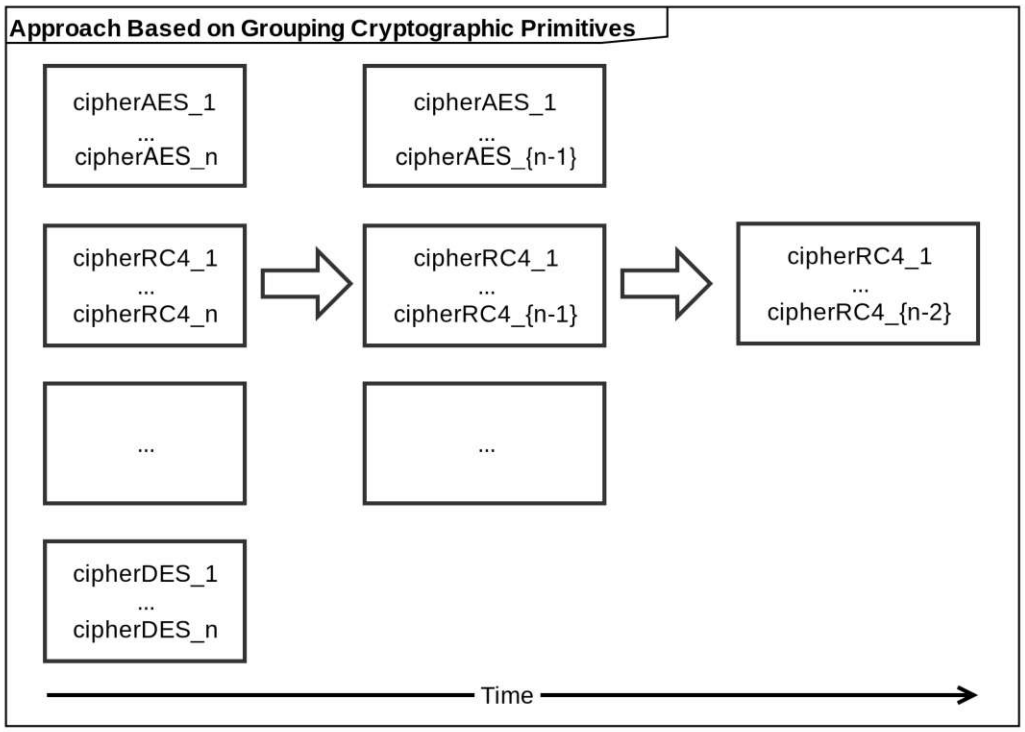


Figure 3.4: Approach Based on Grouping Cryptographic Primitives

Based on Existing Results The third approach, as presented in Figure 3.5, combines the ideas from the former approach with data from already conducted cipher suite scans. It is based on the fact that many server operators use the same configuration, e.g., a default configuration. The most likely configuration based on former results is calculated before a `client_hello` is sent. After the first round of concurrent handshakes, an intermediary result is evaluated. Based on this result, the next, most probable configuration is computed. The cipher suites used in the next round of parallel sent `client_hello` messages are then adjusted. This goes on until all cipher suites are either rejected or accepted. This approach is based on data described in the next paragraph.

Existing Data For the last algorithm, we rely on the dataset of an Internet-wide study we conducted from April to August 2015 [3]. We additionally use cipher suite scans of the HTTPS ecosystem, performed in August 2015. These datasets are very extensive w.r.t. the number of scanned cipher suites. Because of the large dataset (approx. 12 million error-free results), we transformed each result for each single host/port combination to

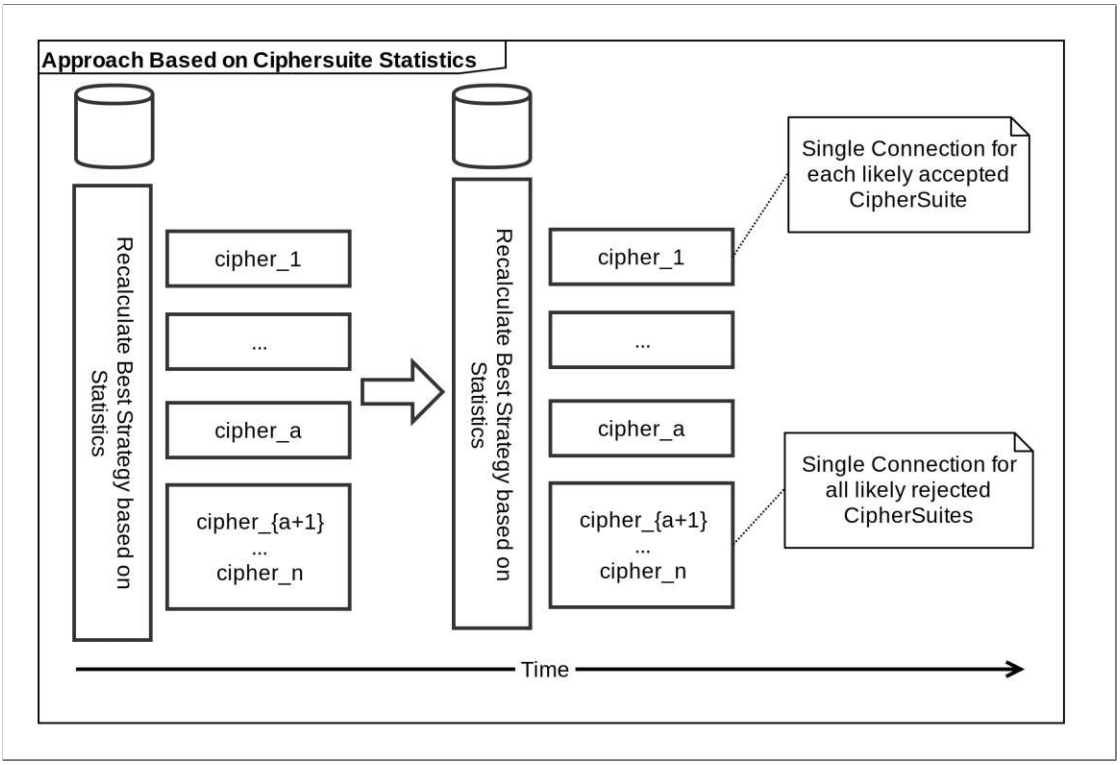


Figure 3.5: Approach Based on Cipher Suite Statistics

a string. This string has a length of 551 characters[9]. This represents the total number of TLS version and cipher suite combinations. Each TLS version/cipher suite is either accepted or rejected, which is represented by the characters a respectively r. We did not take different behaviors of key exchange algorithms or different error messages (for rejected cipher suites) into account. In Table 3.5, the five most-used combinations for HTTPS are shown (a black bar represents an accepted cipher suite, a white bar a rejected cipher suite). We see that 7.8% of all hosts share one configuration in which all cipher suites for SSL 2 and SSL 3 are rejected and the supported cipher suites for TLS 1.0 and TLS 1.1 are identical. As an example, the first two bars represent the accepted `AES128-SHA` and `AES256-SHA`, whereas the next cipher suites are rejected (`CAMELLIA128-SHA`, `CAMELLIA256-SHA`).

When we take a closer look at the number of existing patterns per TCP port and the percentage of hosts that use these patterns, we see that a small number of patterns are used for most of the hosts. This is especially true for SMTP, where we see that the two most-used patterns cover more than 50% of all SMTP-enabled hosts. In Figure 3.6 the

[9]551 cipher suites were tested with *SSLyze* version 0.11. Because the underlying TLS implementation changed, version 0.12 does not test two specific cipher suites for four TLS versions, thus only 543 connections. Existing results for these cipher suites are ignored in the algorithm.

7.8%	
5.5%	
5.3%	
4.3%	
2.7%	

Table 3.5: Most-used cipher suite patterns for HTTPS, Internet-wide scan in Aug. 2015

percentage of hosts that is covered by an increasing number of patterns for various TCP ports is shown. We assume that it is possible to optimize scanning methods by using this information.

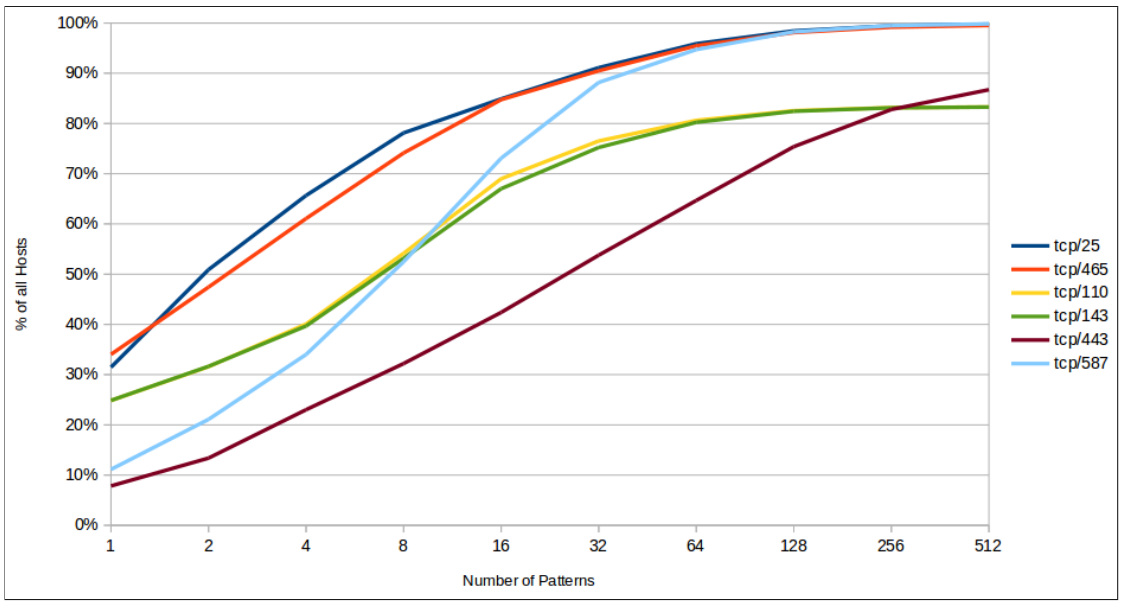


Figure 3.6: Host Coverarage By Number of Patterns

Implemented Approaches

We implemented all approaches by creating an additional mechanism to store partial results. Based on this partial result, the next requests are computed and executed, adding information to the partial result until it is complete. Users are able to choose the algorithm by specifying a command line argument (e.g., `-algorithm=connopt`). The required connections and time are logged for every run. Based on the existing data, we also implemented a simulation that calculates the number of necessary connections and rounds per approach. The complete source code is publicly available[10].

[10] `https://github.com/WilfriedMayer/turning-active-tls-scanning-to-eleven`

	Port 25		Port 110	
	C	R	C	R
Naive	551.0	1.0	551.0	1.0
Connection-optimal	110.0	22.0	42.7	8.5
Crypto-group-based	252.0	7.3	199.9	3.8
Existing-data-based	141.0	1.5	52.3	1.5

Table 3.6: Comparison of Simulation Results with Existing Scan Data

	Port 143		Port 443	
	C	R	C	R
Naive	551.0	1.0	551.0	1.0
Connection-optimal	42.7	8.5	28.2	5.6
Crypto-group-based	199.8	3.8	187.7	2.4
Existing-data-based	51.4	1.4	37.3	1.8

Table 3.7: Comparison of Simulation Results with Existing Scan Data

3.2.2 Results

We evaluated the proposed improvements by simulating an Internet-wide scan on IPv4 with existing scan data. We computed two performance values: The number of average established connections necessary to scan one host (C) and the number of average rounds (round trips) to scan one host (R). These two parameters are a good indicator for the defined requirements. Generated traffic and connections are directly mapped to C, the degree of parallelization and therefore the time needed is mapped to R. The results of this simulation are shown in Tables 3.6, 3.7. We can see that all new approaches use fewer connections than the naive approach. The optimum is achieved with the connection-optimal approach, although this method uses a lot of rounds and is therefore not parallelizable (and probably the slowest of all algorithms), except of the different TLS versions. Thus, the number of connections is five times bigger than the number of rounds. The group-based algorithm lies in between, with further potential to optimize the chosen groups. The algorithm based on existing data shows a low number of connections as well as a low number of rounds. For HTTPS on port 443, it minimizes the number of connections to an average of 37.3 (6.8%) with an average of 1.8 rounds. The simulation is based on the same dataset as the algorithm, so the expressiveness of this method will decrease in the future (as configurations change), but can be easily readopted with newer results.

Experimental Results

We tested the performance of our algorithms with scans in the wild. We used *SSLyze* version 0.12 and scanned a predefined set of hosts out of the Alexa Top10k list. We shuffled it and created batches of 100 hosts. With each algorithm we scanned 25 batches

Approach	# Scans (Hosts)	Valid Results
Naive	25 (100)	1,866
Connection-optimal	25 (100)	1,896
Crypto-group-based	25 (100)	1,914
Existing-data-based	25 (100)	1,870
Connection-optimal	5 (2,000)	9,262
Connection-optimal	5 (2,000)	7,534

Table 3.8: Experimental Results of the Different Approaches (Overview)

	Time (s)			
Approach	Total	Min	Avg	Max
Naive	14,356	0.92	7.69	15.42
Connection-optimal	4,473	0.92	2.36	4.70
Crypto-group-based	5,462	0.64	2.85	4.96
Existing-data-based	4,672	0.50	2.50	5.98
Connection-optimal	5,951	0.56	0.64	0.75
Connection-optimal	9,493	1.13	1.26	1.38

Table 3.9: Different Approaches (Time)

and measured the time needed and connections performed. We refrained from changing other aspects of *SSlyze*, like multiprocessing, multithreading or the general process. We also did not optimize kernel settings or other parameters on operating system level in order to compare only the algorithms with the default behavior. We used commodity hardware with an 100MBit/s uplink. The results are presented in Tables 3.8, 3.9, and 3.10. The naive approach performs worst in terms of speed. Also, connection-wise every new approach performs better than the naive approach. Although it has a more complex implementation, the approach based on existing data performs only slightly better than the algorithm based on crypto groups. Also listed in Tables 3.8, 3.9, and 3.10 are scans with a slightly larger set of hosts, used in the Section 3.2.2.

Figure 3.7 visualizes the large performance gain we can achieve with our approaches. It shows the average time for one host and the average number of connections per valid, scanned host of every tested batch.

These results show a large improvement in TLS cipher suite scanning algorithms. The connection-optimal algorithm is 3.2 times faster than the naive implementation (avg. connection-optimal compared with avg. naive) and uses only 6% of the connections (avg. connection-optimal compared with avg. naive) to execute a full TLS cipher suite scan in the wild. The connection-optimal approach and the group-based approach are correctly simulated, but we see that the results of the method based on existing data differ from the simulated results. We argue that this is due to two reasons: First, the

Approach	Connections			
	Total	Min	Avg	Max
Naive	1,012,976	542.6	542.9	543.0
Connection-optimal	60,723	28.7	32.0	34.7
Crypto-group-based	351,534	182.1	183.7	185.8
Existing-data-based	268,814	126.1	143.8	156.4
Connection-optimal	314,398	33.3	33.9	34.4
Connection-optimal	244,644	31.7	32.5	33.5

Table 3.10: Different Approaches (Connections)

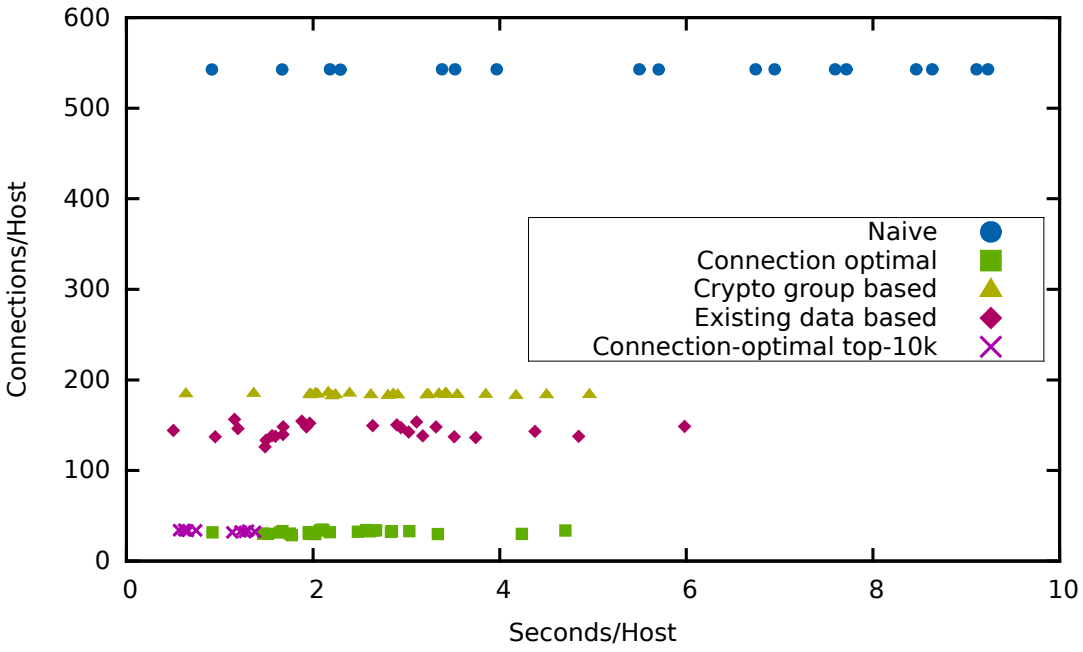


Figure 3.7: Experimental Results of Different Approaches

algorithm and the simulation are based on the same data. If configurations change, the algorithm gets slower. The second reason is that we practically evaluated top-10k web services and not random hosts, whereas the simulation also considers a large number of small hosts.

Cipher Suite Results of Top-10k Domains

We used the connection-optimal algorithm to perform an additional cipher suite scan on the Alexa top-10k domains [59]. Cisco Umbrella recently proposed an alternative to the often-used Alexa Top 1 million list [144], so we decided to scan these top-10k domains as well. We analyzed which patterns occur, if these patterns are secure and

	Umbrella Top10k		Alexa Top10k	
e.g., xx.fbcdn.net	18.53%	1716	8.51%	641
e.g., google.com	13.43%	1244	6.15%	463
e.g., configuration.apple.com	7.63%	707	1.87%	141
Mozilla Modern Conf.	0.02%	2	0.05%	4
Mozilla Intermediate Conf.	0.98%	91	3.28%	247
Mozilla Old Conf.	0.35%	32	0.15%	11

Table 3.11: Cipher Suite Patterns

if we can find a trend to common and secure TLS configurations. First, we looked at the most-used patterns in the Umbrella top-10k list. Although the three most-used patterns are used by 39.6% of the Umbrella top 10k resp. 26.9% of the Alexa top-10k, 524 and 954 (Umbrella/Alexa) different configurations exist. This indicates a highly diverse ecosystem. Second, we analyzed proposed cipher suite settings. Mozilla introduced a tool, the *Mozilla SSL Configuration Generator* [145], to generate secure configurations for various compatibility requirements, i.e., modern, intermediate and old. We see that the cipher suite pattern for a modern configuration is only used by 2 resp. 4 hosts in the top-10k lists. Their intermediate configuration is used by a recognizable number (91, 247). The exact numbers are also shown in Table 3.11. Third, we looked at differences between these patterns. All patterns disabled SSL 2 and SSL 3. In contrast to the modern Mozilla configuration (only TLS 1.2), the other configurations support TLS 1.0 to TLS 1.2. In contrast to the intermediate configuration, Tripple Data Encryption Standard (3DES) with Diffie–Hellman (DH) key exchanges is not supported. The *xx.fbcdn.net* configuration is supporting more cipher suites (CAMELLIA, non-elliptic-curve Diffie-Hellman), whereas configurations like *google.com* support only one cipher suite more than configurations like *configuration.apple.com*, i.e., `AES256-SHA`. Finally, we tried to compare the results with pattern statistics we used for our simulation. We see that there are differences in the pattern usage, and we argue that the average top-10k host is differently configured than the average host from an Internet-wide scan.

3.2.3 Discussion

Internet scanning is not only a technical challenge. It also has to deal with ethical issues. Other studies already pointed out current best practices [42] which include to "scan no larger or more frequent than is necessary". This discouraged studies from performing a full scan, e.g., Holz et al. [53]. They stated that a full TLS cipher suite scan "is a poor trade-off in terms of good Internet citizenship versus lessons that can be learned". With our work, full TLS cipher suite scans can be conducted with less than 6% (approx. 32) of the connections compared to the currently used naive algorithm (543 connections). This minimizes the load each target host has to handle to a manageable minimum and makes the trade-off in terms of good Internet citizenship arguable. *Good Internet citizenship* is not only about minimizing the impact of one scan. It is also about avoiding unnecessary

scans at all. One solution are publicly available results of scans, for which *Censys* [56], a search engine for Internet-wide scans, is a good example. They use the scanning approaches mentioned in Section 3.2.1, but an integration with the results of full TLS cipher suite scans is possible.

In this section we optimized the methodology for full TLS cipher suite scans. For the practical evaluation we didn't change important factors of *SSLyze* to speed up the process. Important factors to optimize the bandwidth usage are, e.g., TCP port reuse, optimal settings for the TCP/IP stack or TCP connection reuse. The most influential factor is the parallelization of the scanning infrastructure. *SSLyze* (version 0.12) uses a maximum of 12 processes with 15 threads for all hosts; if the number of hosts is larger than that, the hosts are queued internally. This behavior is not optimal, since the server is idling. The solution is to split up all hosts amongst a large number of concurrent processes to minimize idling. With some optimizations applied, we were able to scan 27K hosts per hour with the naive approach on commodity hardware (100MBit/s uplink). We did not bundle these optimizations with our new approaches in order to focus on our comparison.

The approaches are created for TLS versions up to TLS 1.2. With TLS 1.3, many things change. Insecure features are dropped, e.g., static RSA or DH key exchanges, insecure ciphers or hash-functions like Message-digest algorithm 5 (MD5). Also, the handshake mechanism changes, so only one round trip is necessary to establish a full TLS connection. This – and also the question how TLS 1.3 is going to be deployed in the wild – affects the problem of how to efficiently scan full TLS 1.3 configurations.

3.3 Securing the Internet, one HTTP 200 OK at a time

This section is based on [6, 7]. The underlying study is also outlined by Krombholz [146].

HTTPS is the most commonly used cryptographic protocol on the Internet. It protects communication content and provides endpoint authenticity at scale. However, deploying HTTPS in a truly secure fashion can be a challenging task even for experienced admins. To explore why this is the case and how these challenges can be fixed in order to support an even wider adoption, we conducted a user study which was presented at USENIX Security [6].

3.3.1 Targeting the long tail

Nowadays, major online services provide TLS encrypted communication. But is the website of your local sushi restaurant secured with HTTPS? Because even your local sushi place needs HTTPS! We need HTTPS as the new standard in the Internet, for all websites, finally deprecating unencrypted HTTP. But often, the opposite is stated. Even at USENIX Security, somebody in the audience questioned the need for HTTPS for small businesses or static content. The answer was simple: more confidentiality, authenticity, and integrity make the web a more secure place. We know that the upgrade to HTTPS is a way – and it is a quite promising way – to improve the security of the entire Internet. And this improvement is strongly needed.

Shifting to HTTPS is not only the problem of single service providers and the process of enabling it is not only a problem of some individuals. It is an enormous challenge for all of us. Developers, administrators, and security researchers are working on it and the situation is improving steadily. We see progress, as Felt et al. [147] showed in their recent work. The number of HTTPS page loads is rising and is now exceeds 50% of strict page loads in Firefox telemetry and similar results with Google Chrome statistics. But their work also identifies weak spots. HTTPS usage on Android devices, the deployments in east Asia, and the long tail of websites are lagging behind. With our study [6], we focus on this long tail of websites that are not supporting HTTPS in contrast to major service providers. These companies – ranging from medium enterprises to your local sushi restaurant – do not have highly specialized security experts in charge, actively checking their websites for improvements. Normal IT departments, single administrators and “IT guys” are solving problems here. And as we already speculate from anecdotes and experiences it is not easy to deploy HTTPS in a secure yet compatible fashion. Probably you remember your first time setting up HTTPS, maybe your last time. Even after initial deployment, the sheer complexity of keeping it secure can overburden experienced admins. Back in 2015, Yan Zhu showed a video with knowledgeable members of EFF, who have never setup HTTPS before. They had troubles deploying it within a limited amount of time.

Enough anecdotes and speculations. We decided to conduct a user study to analyze these problems. We did this without automated tooling provided by *Let's Encrypt* and chose the traditional approach of using a non-automated CA to issue certificates.

3.3.2 User study - Technical measurements

For the qualitative exploration of the usability of HTTPS, we conducted a lab study with 28 participants. Furthermore, we assessed the final configurations quantitatively. We used a common deployment scenario: a Linux operating system, an Apache web server, and a mockup of a simple Certificate Authority (CA). The fictive CA was available through a simple web interface and required the submission of a valid Certificate Signing Request (CSR) for issuing a valid certificate. Additionally to the final configuration files, we analyzed the command line (bash) history and all visited webpages (browser). We evaluated often used commands and parameters, the sequence of executed commands, common webpages, and executed search keywords. More interestingly, we graded the final configurations with evaluation criteria based on the Qualys SSL Test [54]. With the qualitative and quantitative results, we were able to derive the workflow presented in Figure 3.8.

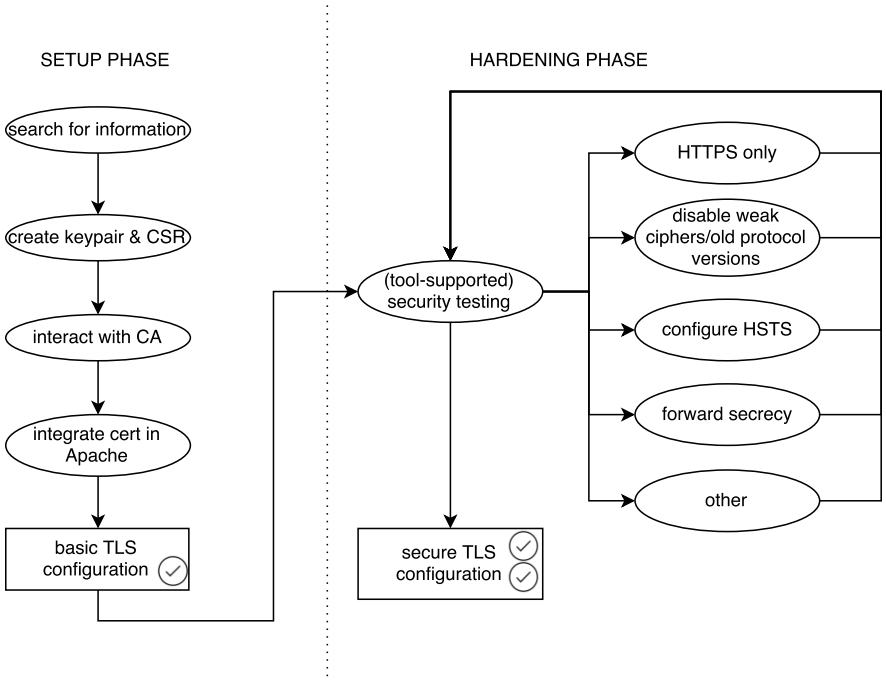


Figure 3.8: Representation of a workflow [6, 7, 146]

3.3.3 Usability Challenges

We identified following usability challenges:

First, searching for information and finding the right workflow. Our participants visited a high number of websites and used multiple sources of information. The average number of visited websites was 60 and the most visited pages were the Ubuntu wiki and the official Apache documentation. We found that the participants jumped between sources that were not compatible to each other and could not assess the correctness of the used sources.

Second, creating a CSR. The creation of a keypair and the CSR is part of the setup phase. We found that many participants had problems understanding the concept and struggled with manually creating the CSR correctly on the first time.

Third, using appropriate cipher suites, that define the underlying cryptographic primitives to be used. A sane configuration defines a limited set of cipher suites with strong authentication and encryption. This is enabled via one specific Apache directive *SSLCipherSuite* with the correct values. However, a deep understanding of the underlying algorithms is necessary in order to make an informed decision. All participants trusted online sources because of their missing knowledge, which implies that the quality of these sources is crucial.

Fourth, Strict HTTPS. We observed that participants were initially confused with simultaneous enabled HTTP and HTTPS. The configuration of strict HTTPS took some time, which could be improved by HTTPS automatically deprecating HTTP.

Fifth, multiple configuration files. All but 6 participants struggled with the configuration file structure, regardless of their experience with Apache. Several participants did not understand how to enable the SSL module or where to configure the entry *SSLEngineOn*.

Sixth, compatibility issues. Participants were unsure about the trade-off between security and compatibility.

Our results reveal that these usability challenges are a serious issue to work on, and that they are the main reason for weak configurations. Fortunately, there are already tools out there. The impact of these tools ranges from generating sane configurations, to comprehensively changing the TLS ecosystem as a whole. We now introduce four tools, that are all tackling these usability challenges.

3.3.4 Tool support

One of the projects with the biggest impact on the TLS ecosystem, especially the long tail is *Let's Encrypt*. It "is a free, automated, and open Certificate Authority.". While the cost-free issuance of certificates takes away any economic reason to not implement HTTPS, and the open design facilitates transparency and continuous monitoring, the automated fashion of *Let's Encrypt* highly improves the usability of certificate issuance. This corresponds to the setup phase (the first column in Figure 3.8) of our identified TLS deployment process model. All setup phase steps are replaced with a single tool-supported step. The certificate issuance is completely automated with the Automatic Certificate Management Environment (ACME) protocol, so no further manual input is needed and

the major web server software, e.g., Apache, is also integrated. *Let's Encrypt* changed the TLS ecosystem significantly, with now more than 100 million certificates issued.

Tool support for the second phase – the hardening phase – is also quickly improving. For choosing the appropriate cipher suites and associated compatibility issues Mozilla published their *Mozilla SSL Configuration Generator* [145]. With selected server software and a chosen compatibility mode, the tool generates a valid and sane configuration. This can easily be copied and pasted into the correct server configuration file. Complicated decisions are replaced with few more easily understandable options. The compatibility level has three profiles: old, intermediate and modern. The tool shows the oldest compatible clients upon selection and also adds correct HSTS headers to the configuration. So it also unburdens the use of HSTS.

Further tool support exists for testing the deployed security configuration. For publicly reachable domains this enables iterative configuration with repeated security testing until a specific grade is reached. The most established testing tool is the Qualys SSL Server Test, the same tool we used to grade the participants' configurations. With Hardenize (hardenize.com) this concept is widened to DNS, email and application security.

In our study, we addressed the ecological validity of our results by conducting additional expert interviews with experienced security consultants. One expert mentioned the web server software Caddy (caddyserver.com) which actives HTTPS per default. We have to shift this paradigm of "security by default" to other major web server software. This tool support – although not yet fully integrated – provides important assistance for administrators, but there is still a lot of work to do, to shift the Internet to HTTPS.

3.3.5 Outlook

As mentioned, the Firefox telemetry data showed that more than 50% of web pages are loaded over HTTPS. This is an enormous success, and the overall trend is definitely pointing in the right direction We see a lot of effort to shift the ecosystem towards HTTPS. With TLS 1.3, not only the security but also the performance of TLS is increased, making HTTPS even more attractive. We see more and more hosting platforms switching to HTTPS for their customers, which is increasing HTTPS usage for the long tail at scale. New upcoming standards, like the ACMEv2 standard [148] are improving the automation of certificate issuance. And the future support of wildcard certificates with *Let's Encrypt* [149] will form the ecosystem even more.

In conclusion, administrators of the long tail of the Internet should sometimes also be seen as users. Some configuration tasks still require a deeper understanding of security mechanisms or even underlying cryptographic methods. If we want these security mechanisms to work, we also have to support administrators in enabling them. With this in mind, we should all work on shifting the ecosystem, until even the long tail supports HTTPS. So that we can finally move on to the Internet where HTTPS is the norm.

3.4 TLScompare: Crowdsourcing Rules for HTTPS Everywhere

This section is an extended version of [8].

One of the major problems (effective 2016) was the lack of wide-spread HTTPS deployment: while it is a well-understood concept and many websites protect portions of their communication content with clients using HTTPS, e.g., login forms, only a minority of websites serve their entire content over HTTPS. While protocol extensions like HSTS [61] can be used to enforce HTTPS for specified domains, only few website operators deployed it [46]. This leads to numerous security- and privacy-related problems, as an active adversary can read, modify or redirect traffic at will during a man-in-the-middle (MITM) attack. This can be either a malicious ISP [133], a local attacker on the same network using Address Resolution Protocol (ARP) spoofing [150], or a malicious CA. In a recent paper it was demonstrated by Gmail that not only HTTPS is commonly targeted by active attackers. Also attacks against other protocols which use TLS, e.g., SMTP are encountered in the wild [75].

Server administrators are responsible for deploying HTTPS for a given website, and there is little that users can do to protect their communication content if servers do not support HTTPS. However, many websites employ HTTPS for a fraction of their pages, resulting in a valid certificate and proper HTTPS deployment – many times serving the same webroot over both protocols. The browser extension *HTTPS Everywhere* redirects otherwise unencrypted communication content to the encrypted counterpart where available, with a set of rules written in regular expressions. So far these rules are manually maintained, and while everyone can create his/her own rules, this approach clearly doesn't scale to the current size of the web. Therefore, we present our platform *TLScompare* in this section, which tries to scale rule-creation and -verification using crowdsourcing.

The remainder of this section is divided into the following parts: Section 3.4.1 introduces our approach for crowdsourcing rule generation. Section 3.4.2 presents our implementation for the collection of these rules. Section 3.4.3 discusses our findings.

The rules for HTTPS Everywhere are written using regular expressions which specify the target domains as well as the secure counterparts that ought to be used instead. Listing 1 presents a simple rule for the domain `torproject.org`. Not only does it rewrite all connections from http://torproject.org to https://torproject.org, but also all valid subdomains with the * wildcard.

These rules are manually developed and maintained using a public git repository. At the time of writing the repository contains approx. 18,400 files that specify rewrite rules. The process of manually creating rules is justified by the possible complexity of rewrite rules for particular deployments, yet simple rewrite algorithms do sometimes not know how to handle these deployments correctly. During the compilation of the browser extension,

[11] `https://github.com/EFForg/https-everywhere/blob/master/src/chrome/content/rules/Torproject.xml`

```
<ruleset name="Tor Project">
  <target host="torproject.org" />
  <target host="*.torproject.org" />
  <!-- excluded content -->
  <rule
    from="^http://([^/:@\.]+\.)?torproject.org/"
    to= "https://$1torproject.org/" />
</ruleset>
```

Listing 1: HTTPS Everywhere Rule for torproject.org[11]

```
<ruleset name="example.org">
  <target host="example.org" />
  <rule from="^http:" to="https:" />
</ruleset>
```

Listing 2: Trivial rewrite rule

these xml files are merged into a single SQLite file. Table 3.12 shows that the majority of the existing rules of HTTPS Everywhere are in fact really simple. 13,871 files contain only one rewrite rule and of all rules 73% do not reference the HTTP URL (i.e., are not using $ in the regular expression). 28% use the trivial rewrite rule "http:" to "https:" as listed in Listing 2 and only 1.6% use a more complicated rule with two references. The "exclusion" element is used very rarely.

Files with 1 rule	13,871
Files with 2–10 rules	4,496
Files with more than 10 rules	44
Total rules	26,522
Trivial rules	7,528
Rules without reference ($)	19,267
Rules with more than one reference ($)	417
Files with no exclusion	17,005
Files with one exclusion	1,136
Files with more exclusion	272
Files with no securecookie	8,597
Files with one securecookie	9,215
Files with more securecookie	601

Table 3.12: Ruleset statistics

This great number of rather simple rules leads us to the assumption that the process of rule development can be automated to some extent.

3.4.1 Design

We split the design of automated rule generation into two parts. First, suitable rules are generated with a rather simple algorithm. Second, these rules are validated by matching the similarity of the encrypted and unencrypted content with crowdsourcing methods.

Automated Rule Generation

Many domains are deployed with a simple HTTPS configuration. For these domains, rules can be automatically generated with tool support. HTTPS Everywhere provides a script to generate trivial rules for these domains (called `make-trivial-rule`). Rulefiles generated with this script include one or more targets and the simple rule as listed in Listing 2. We can identify hosts that support HTTP and HTTPS via Internet-wide scanning techniques. Hosts that use HSTS or server-side redirects can also be efficiently identified and excluded from further processing. We then use a rather easy approach of rule generation by creating trivial rules for all domains. We also include often-used subdomains such as "www" and "secure". Also, the rank from the Alexa Top 1 Million ranking [59] is taken into consideration.

The most important requirement for new HTTPS Everywhere rules is that we want to avoid corrupt rules under all circumstances. If error-prone rules reduce user satisfaction, the plugin may get uninstalled. Of course the assumption that two pages serve the same content is naive, although often true. If we want to avoid corrupt rules, we have to further prove the similarity of encrypted and unencrypted content.

Rule Validation

The question whether the content of two pages is similar is not easy to answer. We can compare the equality of two pages by comparing their hash values, but if, e.g., only the server name differs in http and https, the pages will not share the same hash value despite being similar. This problem continues with dynamically generated content. The pages can be similar, indicating a valid rule, but they are not equal, e.g., if they display different ads or other dynamically loaded content. To measure the similarity and thus conclude if a rule is valid or not, we identify two possibilities: First, similarity-matching algorithms. These algorithms can be based on the HyperText Markup Language (HTML) source code, visual recognition, or other metrics. Second, crowdsourcing approaches, i.e., we use the input from a large group of people who voluntarily compare pages. To facilitate this approach we developed a web service that is easy to use and publicly available. We can then use crowdsourcing results to measure the correctness of rules and analyze the reasons why rules are invalid.

3.4.2 Crowdsourcing

In order to crowdsource the validation of rules, we created the project *TLScompare.org*. This web service facilitates the rewrite rule validation process by providing an easy-to-use interface. As illustrated in Figure 3.9, this minimalistic interface offers three buttons: one to start the comparison and two to choose the result. After clicking the button "Compare!" two separate browser windows appear. The left window displays the non-secured (HTTP) version of a web page, the right window displays the secured (HTTPS) version. These two different URLs represent the "from" and "to" value of the regular expression. The browser windows are opened without a menubar or a scrollbar. The windows are also equally sized and do not overlay the buttons of the main window. After comparing the two pages the user has two choices in the normal operation mode. This is illustrated in Figure 3.10.

Not equal The two pages differ in some way.

Equal The two pages are visually equal.

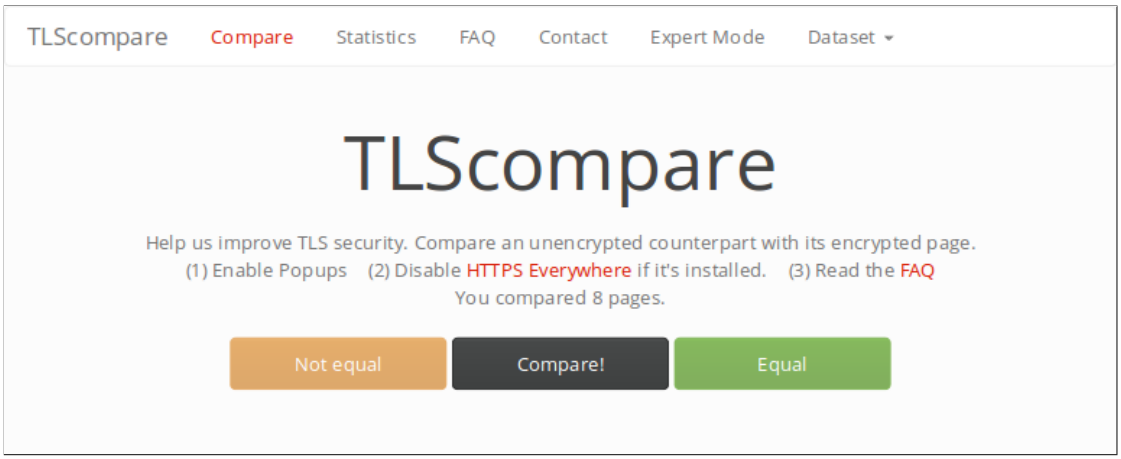


Figure 3.9: Normal operation mode

The meaning of these choices is explained in the FAQ section. After the user chooses one result, both windows close automatically and the next comparison can start.

As shown in Figure 3.11, we also introduced an "expert mode" to analyze the differences on a more fine-grained level. The expert mode enables a variety of result choices, specifying the detailed reason. Possible choices with short explanations – also shown in the corresponding tooltip – are:

Certificate Mismatch Certificate URL is different to URL.

Untrusted Certificate Expired certificate, or untrusted by browser.

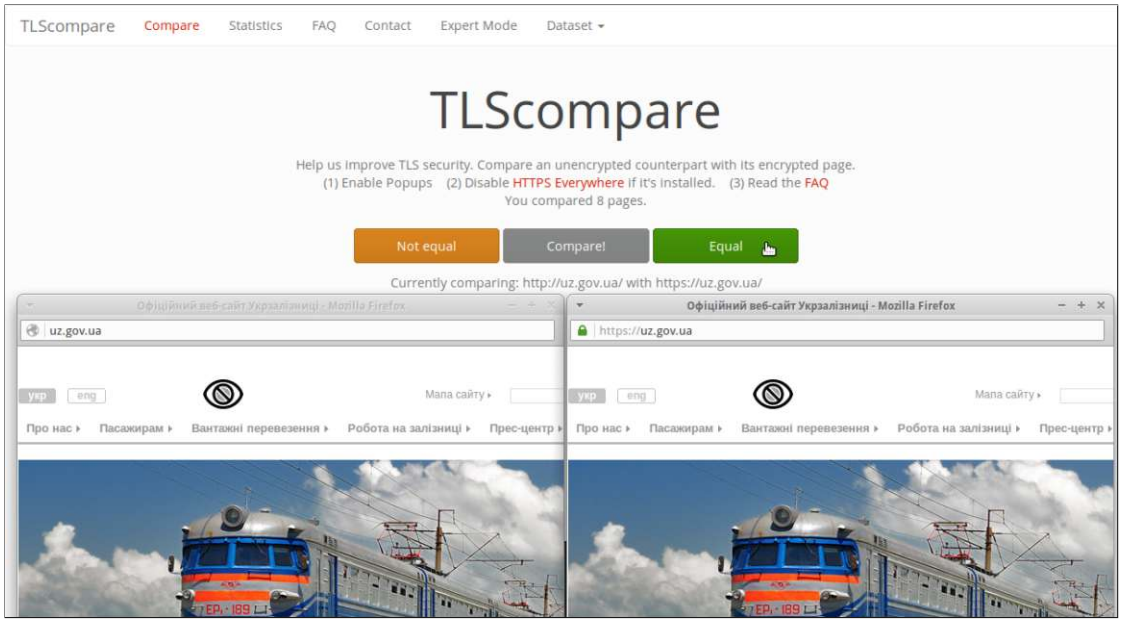


Figure 3.10: Comparison screen

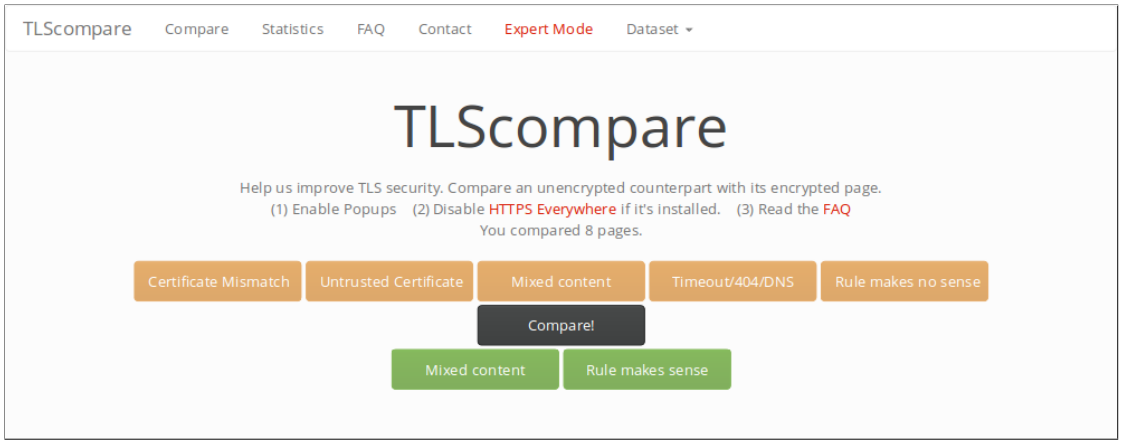


Figure 3.11: Expert operation mode

Mixed content Pages do not look similar due to mixed content (Cascading Style Sheets (CSS), JavaScript (JS)).

Timeout/404/DNS Network related errors on either page.

Rule makes no sense No use in upgrading: pages are not similar or either page redirects, e.g., from HTTP to HTTPS (or vice versa).

Mixed content Pages look similar, but mixed content warning.

Rule makes sense Pages look similar, no HTTPS warnings.

The user is able to compare pages for different datasets, while we are able to set different default datasets. We also included an administration panel as illustrated in Figure 3.12.

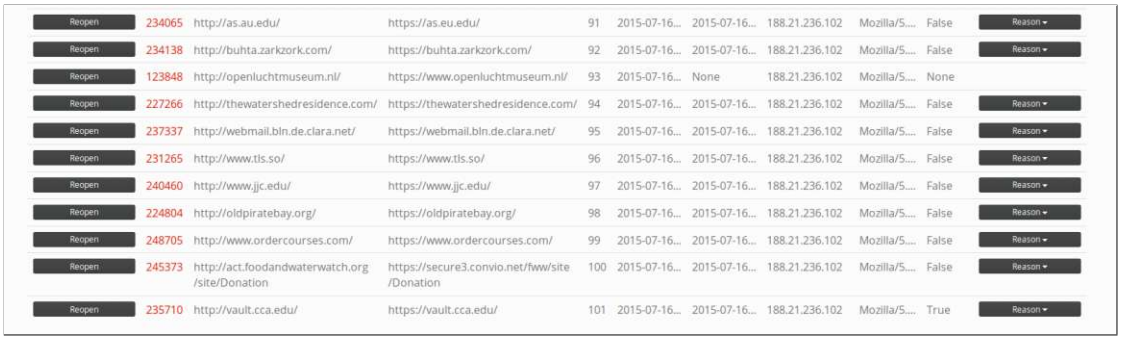


Figure 3.12: Administration panel

Technically TLScompare is developed as a Python Web Server Gateway Interface (WSGI) application behind a nginx webserver. Flask [151] is used as web framework and the results are stored in a relational SQLite3 database via SQLAlchemy [152]. All important methods are exposed via Representational state transfer (REST) endpoints. This setup is hosted at `https://tlscompare.org`. For every result we store the IP, the User Agent, the result, the reason (if classified in expert mode) as well as the timestamps of the original request and original result. Each validation attempt is identifiable via a unique id and connected to a session identifier. We are able to filter out bogus results via wrong session identifiers and a too short duration between request and result.

To improve the user motivation we included basic statistics for the current session and the global scope as depicted in Figure 3.13. The user is able to compare the number of results to the global scope or other users.

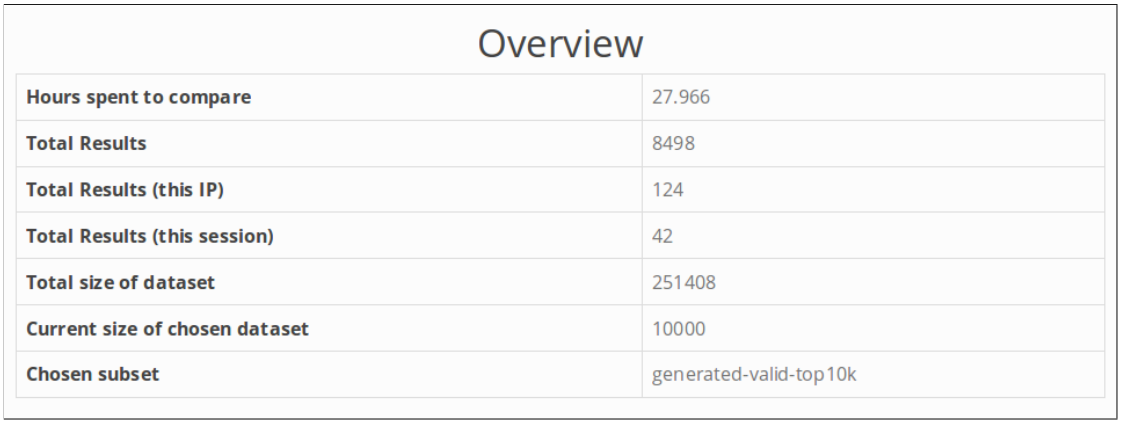


Figure 3.13: Statistics on global and session scale

Acquired Data

At the time of writing we have acquired 8,523 validation attempts and 7,616 actual results. An overview of the acquired data is presented in Table 3.13.

Unfinished attempts	907
Actual results	7,616
Number of different UserAgents	125
Number of different sessions	268
Min results per session	1
Max results per session	456
Average results per session	28.4
Median results per session	6
Sum of time between request and result	27.98h

Table 3.13: Results statistics

We introduced different datasets from which to choose:

Existing rules Comparisons of unencrypted pages that are currently rewritten by HTTPS Everywhere. We computed these URLs by "reversing" the regular expressions of rules. For simplicity we only generated rules out of regular expressions that include different subdomains, but don't contain more complicated regular expressions.

Generated rules We also generated values to compare domains of the Alexa Top 1 Million domain ranking. We used comparisons for second-level domains and for common third-level domains, e.g., www, secure, or ssl. We also filtered out a subset with the 10,000 highest-ranked domains.

We expect these datasets to yield completely different values. For the existing rule set, we expect the majority of results to have similar pages. Statistics for this dataset are presented in Table 3.14. We see that there are, against our expectations, more "Non Equal" results. We therefore checked all 226 "Not Equal" results that were set without a reason. We found that 106 of these domains deployed a default server-side redirect and were set before we changed "Not Equal (Rule makes no sense)" in expert mode to explicitly save this reason. We assume that these results were generated by users that realized that additional client-side rules make no sense although the pages look similar. The remaining results must be checked separately.

We also created a dataset of pages for the Alexa Top 1 Million ranking. We used a rather naive algorithm to create new rules which were then compared. The results of this dataset presented in Table 3.15. We see that 74% of all rules were marked with "equal". This number is conform to our assumption that many hosts serve similar pages on the encrypted and unencrypted channel.

Total attempts	543
Total results	527
Equal	169
Not Equal (Total)	358
Not Equal (Without reason)	226
Not Equal (Timeout)	58
Not Equal (Certificate Mismatch)	40
Not Equal (Untrusted certificate)	14
Not Equal (Rule makes no sense)	15

Table 3.14: Dataset for existing HTTPS Everywhere rules

Total attempts	2,600
Total results	2,267
Equal	1,688
Not Equal	579
Not Equal (mixed-content)	87

Table 3.15: Dataset for similar Alexa Top 10k domains

Table 3.16 presents the validity results per created rule. Each "1" represents one comparison where the pages were marked as equal, each "0" represents a non-equal result. It also shows comparisons where more than one result was submitted. For comparisons with two results, 87% have the same result and 13% differ in the result. Apart from users that submitted wrong results, this can have multiple reasons, e.g., the pages look different in some browsers or the similarity changed over time. For comparisons with three results, 86% have the same result and 14% have one discordant value.

Deriving rules

With this data we can automatically generate HTTPS Everywhere rules from results that have been checked multiple times and yielded the same result. In the future, we have to find the correct threshold; currently we assume that comparisons that yield the result "Equal" for at least three times can be taken for sure. We currently have 38 results that fulfill this requirement. The creation realized by selecting all candidates, passing them to a generation script and creating rules as presented in Listing 3, where *BASEADDRESS* is the domain name of the unencrypted part, and *ENCRYPTED* is the domain name of the encrypted URL. These rules are then ready to be inserted in the HTTPS Everywhere repository.

3.4.3 Discussion

We started our project as non-commercial, although several commercial crowdsourcing frameworks exist. The probably most famous variant is Amazon Mechanical Turk [153].

Combination	Number of Results
0	842
1	443
00	612
10	148
11	394
000	67
100	6
110	10
111	32
0000	9
1000	2
1100	2
1110	1
1111	5
00000	1

Table 3.16: Multiple results to ensure data quality

```
<ruleset name="BASEADDRESS">
  <target host="BASEADDRESS" />
  <rule from="^http://BASEADDRESS"
        to="https://ENCRYPTED" />
</ruleset>
```

Listing 3: Trivial rewrite rule

It commercializes so called Human Intelligence Tasks (HITs). Each HIT is solved by a human worker who gets paid with a small amount of money. One page comparison can be modelled as HIT, and with this platform we could easily enlarge our dataset. With our non-commercial project we were able to collect 7,616 results in a timespan of approx. 5 months. We announced our project solely on Twitter and held one lightning talk at a community-driven event. We assume that the quality of our data is comparable to data from other crowdsourcing frameworks. We also assume that we would have to include more sophisticated methods to ensure data quality if we also included money-driven results with commercial crowdsourcing providers.

To further increase the usage of encrypted connections for data transmission, other problems have to be solved first. This includes an easy way to deploy server-side HTTPS and ensure trust in the used certificates. A promising project currently dealing with this issue is *Let's encrypt* [154]. Server-side redirection is making the effort of client-side rewrite rules worthless, and together with the server-side enforcement of HTTPS via

HSTS [61] it is clearly the favorable solution.

Future Work

We currently investigate the use of different algorithms to automatically measure similarity. We have to identify usable algorithms, but the question of similarity in the context of HTTPS support is not easy to define. Can an URL be rewritten to its encrypted counterpart if the content is not equal? If some content of the page is randomly displayed, but the functionality stays the same? Is the content similar if the code is similar or if the rendered images are similar? Different technical methods are possible: simple word matches, comparisons of the Document Object Model, algorithms that compare images of rendered pages amongst others. However, similarity matching algorithms will also yield false positives or create borderline cases. TLScompare can then be used to crowdsource fine tuning of these algorithms or crowdsource manual checking of the results.

We will also adopt TLScompare for a student exercise. After students manually create new rules for HTTPS Everywhere, TLScompare will be used to validate these rules.

Weighting of user quality is currently left out, but is another measure to ensure data quality. Users are not identifiable, but the results are stored with a session id, the User Agent and the IP address. So if one user misjudges more results, the weight of this session is reduced accordingly. Also, the average compare time, i.e., the duration between request and result, is one property that can be used to calculate the user's credibility.

Another future issue is to optimize the selection algorithm for the next comparison. An improved version can immediately choose the same comparison for another user to increase comparisons with more than one result. This is especially true for larger datasets where the probability of multiple results is low.

CHAPTER 4

Measuring Network Layer Privacy

While Transport Layer Security is a major step forward to protect users' privacy by encrypting the content in transit, the implications of metadata are often underestimated. Metadata includes the timing, source, and origin of data, and with enough metadata, information about individuals can easily be derived. On the network layer, many parties are involved in communicating and transferring data. These parties can be seen as potential middlemen. While malicious parties got attention from the research community, so-called trusted third parties, e.g., Internet Service Providers, are often understudied. ISPs could act as potential malicious actors, and also, when they are not deliberately violating privacy, techniques that are used for other reasons could harm privacy as well. Users are often unaware of such threats.

This chapter investigates techniques used for network management and traffic differentiation that could potentially be a threat to privacy. Section 4.1 describes the long-term evaluation of net neutrality measurements within one country with a handful of providers. After conducting the study, we identified the study setting as limited for the ongoing shift to mobile Internet providers and mobile devices. Consequently, Section 4.2 describes a framework to measure mobile providers with a focus on scalability and with an international scope. We showcase the capabilities of such a measurement framework and techniques that potentially harm users' privacy.

4.1 A Framework for Monitoring Net Neutrality

This section is based on [9] and focuses on the long-term evaluation. The fundamental design of the framework and first measurements are described by Schreiber [155] in his master thesis.

ISPs can discriminate traffic in certain ways, e.g., by the used protocol or application. This leads to advantages for providers, but also harms the freedom and innovation in the Internet. However, ISPs currently use a variety of technical measures. Some can be seen as questionable regarding net neutrality – an important topic in legal and economic discussions. These methods are often neither technically identified nor continuously monitored, which prevents an informed discussion about the legitimacy of such methods.

4.1.1 Introduction

Net Neutrality is a highly discussed topic from a technological, economical and legal viewpoint [156–159]. In short, it describes that ISPs should not treat traffic differently based on any characteristic [160], e.g., the protocol used. While legislation, e.g., in the EU or the USA, as well as economic discussions often gain attention, technological discussions of used techniques are scarce. Our study focuses more on these technical aspects. Therefore, we create a specialized measurement platform that is capable of detecting the usage and the consequences of miscellaneous network management techniques. We implement a variety of test cases and conduct three measurement periods of three months each. We then analyze the measurement results and find questionable methods in use. In this section, we solely focus on the technical part and do not discuss the findings from a legal or economic viewpoint.

Techniques used to differentiate traffic and potential net neutrality violations are often hard to detect. Customers sometimes have knowledge of different infringements, but they just communicate it over bulletin boards or directly to network regulators. However, evidence from an integrated and consistent monitoring system is missing. With our work, we introduce a framework that consistently monitors different Internet products with specialized metrics in a transparent way. That is not an easy task, since questionable techniques do not necessarily differentiate from network management methods or security measures. Port blocking techniques are also used in simple network firewalls, changed DNS responses are used in censorship systems and proxying of traffic is often enforced by company policies, but all three techniques are questionable in terms of net neutrality

Because the topic of net neutrality is rather young, measurement studies and frameworks focusing on it are scarce. Related research in the field of Internet measurements exist, e.g., the RIPE Atlas network. In addition, studies from the field of censorship detection as well as methods from intrusion detection systems are relevant. Still, not all these approaches are suitable for our case. We introduce a methodology that enables us to fully control both endpoints and conduct extensive tests in terms of computation and

network bandwidth. This system is scalable in terms of metrics used and in terms of the Internet products checked. One main challenge for a monitoring system is the need for reproducibility and traceability. Simple reports of found techniques have to include a traceback to the original traffic. To facilitate this process, we capture the raw traffic on server- and client-side and link it directly to the generated results.

We then use this framework to conduct measurements for five different ISPs in Austria. We repeat this three-month monitoring process in total three times to validate our results over time. We identify some questionable techniques that are further described in the paper.

The main contribution of this section is:

- Results of three measurement periods (each three months) in Austria, revealing questionable techniques of ISPs and their changes made over time.

The remainder of this section is structured as follows: First, Section 4.1.2 briefly describes the measurement framework, the defined metrics and how the measurement was conducted. An evaluation of the long-term results follows in Section 4.1.3. We discuss these results in Section 4.1.4.

4.1.2 Methodology

We decided to create a client/server-based framework to conduct our measurements. The architecture, as shown in Figure 4.1, utilizes one highly capable server and a customizable number of configured clients. These clients are connected to different Internet products of different ISPs. Although the architecture of this framework seems rather simple, other approaches were just not suitable, as we want to test technical measures within one country. While crowd-sourced and highly distributed approaches like RIPE Atlas[1] or software measurement tools have their clear advantages, our approach enables a fully controlled, fully configurable environment for both endpoints. This allows us to conduct more complicated tests with higher bandwidth usage and higher processing complexity. We conduct all measurements on an hourly basis. That enables measuring changes happening during a day, e.g., different treatment of data at peak times or capturing the exact moment of changes in the use of different techniques

Clients request the planned tests via a REST application programming interface (API) from the server. The tests are specific to different, exchangeable and extensible metrics. The server schedules the tests of the different metrics in a non-overlapping way and returns the upcoming test configurations as JavaScript Object Notation (JSON) objects.

[1] https://atlas.ripe.net/

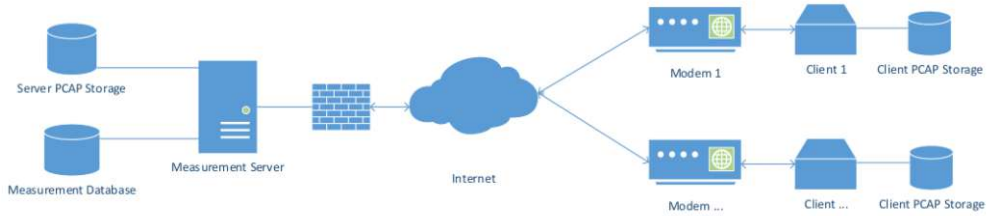


Figure 4.1: Basic measurement framework architecture (Schreiber [155])

Then, the test clients start the measurement at the given time interval. In this time interval, the needed functionality is loaded, the firewall permits only access for the needed ports and the actual test is performed. The results of each test are summarized to a single JSON object (which contains basic information about the test and additional metric-specific results), transferred to the server and stored in a MongoDB database for later analysis. To keep the results traceable, a packet capture (PCAP) file of every test is recorded on the clients and the server.
The source code is openly available[2].

Experimental Deployment

We decided to support five different connections of major Austrian ISPs to test current technical measures occurring in Austria. We included Asymmetric digital subscriber line (ADSL), cable and three stationary Long-Term Evolution (LTE) products into our study. Each connection used its own client hardware and modem. Therefore, we deployed our measurement framework with five different clients. For the client hardware, we used Intel NUCs (2x1.8GHz CPU, 16GB RAM). In comparison to test clients of other measurement architectures that use single-board computers, these rather strong clients did not limit computational intense tests. We were also able to rule out any result noise, caused by undersized hardware (happened, e.g., with the first generation of RIPE Atlas probes). The server was configured with a static IPv4 address and connected via Gigabit local area network (LAN) from the university network of TU Wien. As modems we mostly used operator provided modems (Huawei E5180 and E5170, TG588v, CNB CH7465CE) and one other device (TP-Link TL-MR6400). All contracts were concluded by private customers, to preclude any privilege based on company-based contracts and reproduce the experience of private customers.

Metrics

We defined 17 different metrics. Each of these metrics is designed to measure a single aspect of possible technical measures. We tried to implement a broad spectrum of tests in

[2] https://github.com/sbaresearch/monitoring-net-neutrality

different ISO/OSI layers. We briefly list the used metrics here, the detailed descriptions are given by Schreiber [155]:

- Voice over Internet Protocol (VoIP) – QoS metrics of VoIP streams.
- SYN Flooding –
- Invalid Syntax – Tests with invalid syntax for POP3, SMTP, and HTTP
- TLS interference – Either with TLS or STARTTLS
- HTTP Manipulation – Either via caching mechanisms or because of virus-related content
- Basic network measurements – Basic TCP, User Datagram Protocol (UDP), and traceroute tests
- Censorship tests – With OONI.
- DNS-based metrics – Testing DNS responses.
- BingeOn-related metrics
- Bandwith tests

4.1.3 Long-term Evaluation

In this section, we describe the results of our measurements: First, the datasets generated, second, special results where we found questionable techniques and third, a detailed overview of the results from all metrics. Last, we explain change over time, as the first and last measurement are more than one year apart.

Dataset

The dataset contains test results of three measurement periods over the time span of three months each.

1. August 2016 to October 2016,
   with 140,000 measurement points
2. February 2017 to May 2017,
   with 140,000 measurement points
3. September 2017 to December 2017,
   with 87,000 measurement points

We conducted these tests for five Internet products from different Austrian ISPs. We tested one ADSL, one cable and three stationary LTE products. All test metrics were conducted on an hourly basis with a manually defined scheduling. There are short time frames where no measurements have taken place, due to different hardware failures, maintenance windows and software updates.

Summarized Results

Although we conducted tests for seventeen different metrics, we only found some questionable techniques. In this subsection we are going to describe the most interesting findings and summarize the results.

We found the following techniques:

HTTP middlebox. We found one Internet product that uses a middlebox for traffic over port 80. We found evidence in more than one test and more than one metric. We measure higher time to live (TTL) values for exactly one port and do not see this behavior for other Internet products. This leads us to the conclusion that only traffic on port 80 is treated differently. Also new HTTP headers are added. We see this in our content manipulation metric for the `Host` as well as the `Connection: Keep-Alive` header. We also see that HTTP syntax errors sometimes prevent transmission to the server and commands written in irregular upper and lowercase (e.g., `CONtenT-LeNgtH`) were exchanged with the correct version (e.g., `Content-Length`). Therefore we assume the complete header is exchanged. In the last measurement period, we were not able to detect any irregularity and no middlebox in use, so we assume the provider stopped this technique.

SMTP middlebox. We found that ISP3 uses a kind of middlebox for SMTP traffic. We observe that SMTP traffic is changed in transmission. Wrong server response codes (Code: 136 instead of 250) appear to cause the middlebox to terminate the connection. Therefore, the last (correct) command, `220 2.0.0 Ready to start TLS` is not transmitted. Instead, the message `421 syntax error (wait for server reponse)` is received at client side. Although a middlebox is in place, we have not found any evidence of STARTTLS-Stripping. In the last measurement period we were not able to detect any SMTP middlebox, but SMTP traffic over TCP/25 was completely blocked.

DNS blockades We tested DNS blockades for various domains and found different behavior of the ISP's default DNS servers. We found that all providers blocked the domains `www.kinox.to` and `www.movie4k.to`. Four providers redirected to a webpage on a webserver in their address space showing a legal explanation of the domain blockade. One provider (ISP5) resolved the domains to `0.0.0.0`. For the domain `www.thepiratebay.se` we found that one default DNS server (ISP4) was not preventing access and was resolving the domain name correctly in the first two measurement periods. This behavior did not change over all measurement periods.

Google Custom Search for Non-existing Domain. For non-existing domains, ISP3 (resp. the default DNS server) redirects to an existing address in the address space of this provider. It shows a Google Custom Search Page, which is preselected with the requested domain. This custom search can be seen as a way to monetize requests as described by Weaver et al. [77]. One DNS server produced a timeout after five seconds, all other providers were returning the NXDOMAIN DNS status code correctly. This behavior did not change over all measurement periods.

We base these insights on detailed results from following results:

- Basic TCP and UDP measurements – First, Port 554 was occasionally treated differently for ISP5. Second, and more noticeable, the TTL on port 80 had a significant difference for ISP4. We also noticed that fixed line providers stayed consistent with their TTL values, whereas mobile providers varied to a greater degree. We could not determine any traffic shaping based on our round-trip time (RTT) measurements. Neither depending on protocol nor on hour of the day.
- DNS tests – The default DNS server of one provider did return `0.0.0.0`, others returned IPs from their address space, showing legal information about the blockade. Over time, one provider harmonized the IPs for the legal information, but principally the results did not change over time. We noticed that one provider was not blocking `www.thepiratebay.se` in the first and second measurement period, although other domains were already redirected to their information page. In the third and last measurement period, we see the start of the blockade. We also discovered different behaviour regarding non-existing domains. One provider resolved the domain to a server with a Google Custom Search in place. Another request stopped with a timeout after five seconds.
- SMTP tests – Four tested providers did not interfere with the SMTP communication. One provider changed the transmitted data.
- Tests based on HTTP – Metrics based on HTTP traffic showed inconsistencies.
- Censorship tests based on OONI – With three OONI testdecks we were able verify the results of other metrics. The results also showed inconsistencies in the transmitted data.

Traffic shaping details. In Figure 4.2 we show that we detected different speed test results over the course of a day, but we cannot find any differences based on TCP port. This is true for all three measurement periods.

Compared measurement periods

We conducted three different measurement periods to validate our results and continuously monitor applied techniques. When we compare the different results, we come to following conclusions:

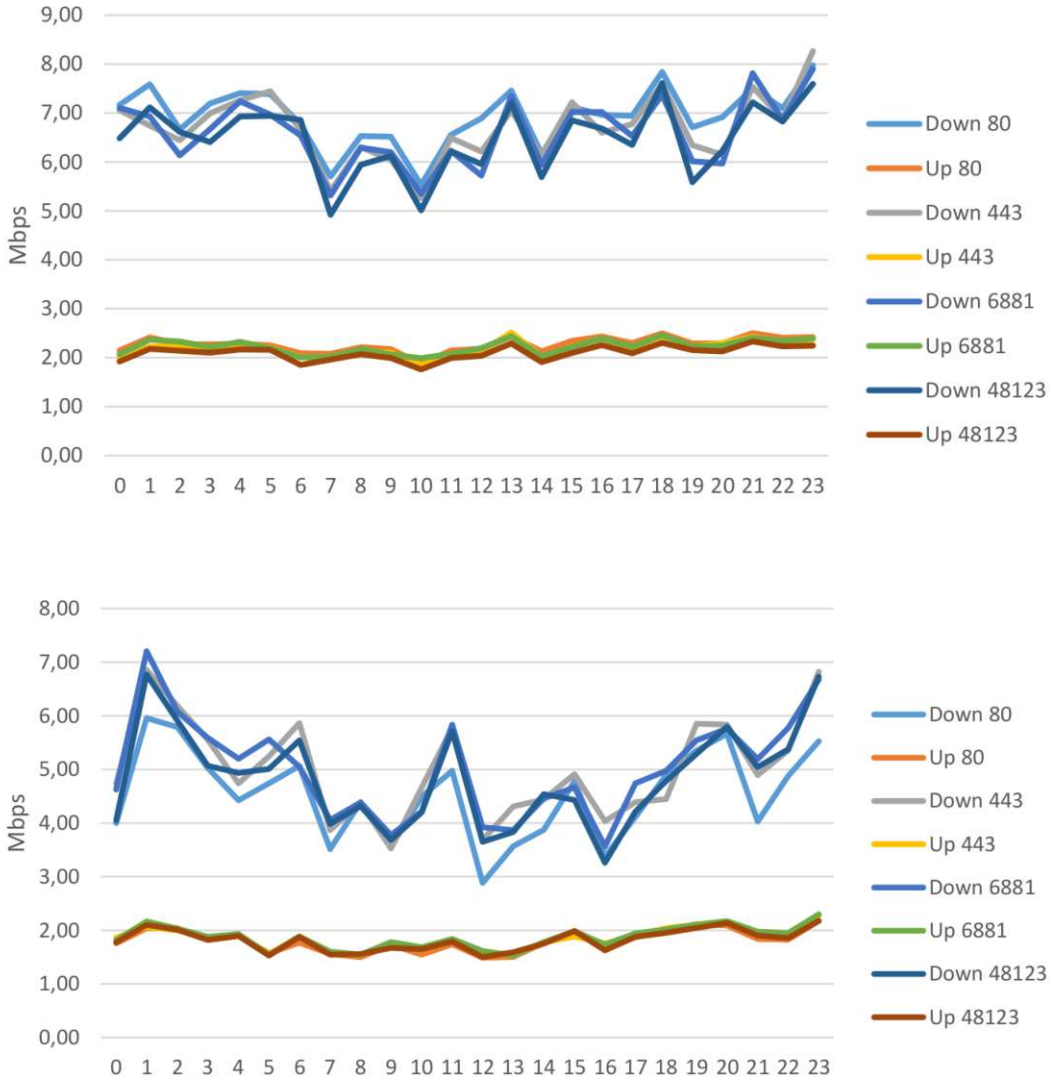


Figure 4.2: Mean bandwidth of various TCP ports for ISP4 over the day, measurement periods: 2 – top, 3 – bottom, measurement points: 2 – n=1894, 3 – n=1342

1. The majority of metrics did not reveal new results. The chosen techniques we identified tend to be rather stable. Single, reoccurring scans seem to be sufficient to detect ongoing usage.

2. Two providers changed their used techniques and replaced their middleboxes. No metric indicated the use of the middlebox on TCP/80. The middlebox on port TCP/25 for SMTP was also removed, but instead a complete blockade was introduced.

Over all three measurement periods, we were not able to identifiy any suspicious behavior for following metrics:

- SYN Flooding
- VoIP
- Email (Client to server via POP3)
- TLS interference
- Caching effects
- Traffic shaping based on TCP port or HTTP header
- Traceroute testing

4.1.4 Discussion

Our results only consider the technical aspects, as they are the only one we can empirically test. We can only speculate about the reasons why specific measures are in place. There are several explanations for different measures. If you take a middlebox for e-mail related ports, you may first see privacy violations but ISPs will maybe argue differently. Security measures or spam prevention are possible explanations. Our framework helps identifying, measuring and continuously monitoring network anomalies, but the consequences still have to be discussed in the concerned field

The effort of ongoing continuous scanning vs. the insights gained can be discussed. We implemented our system as an ongoing monitoring framework for net transparency, but we found out that most techniques are rather stable. For some metrics, scanning on an hourly basis is not necessary, e.g., DNS behavior, were a reduction to daily scanning is still sufficient to monitor changes in a timely manner. Other metrics, e.g., bandwidth tests could be performed more often to gain a deeper insight. We decided to remain our test configuration unchanged throughout the study to increase comparability of the results. You could further argue that more complex metrics could be executed to detect clearly questionable techniques like middleboxes or proxies, while more often executed

ones give us a more detailed picture.

With the 17 implemented metrics, we tried to cover a wide spectrum of possible technique detection. Still, our work is limited by the number of possible tests. With releasing the source code, we encourage other researchers to implement their own metrics for continuous net transparency monitoring. In addition, metrics could be improved in various ways, e.g., full protocol simulations instead of simple port-based traffic shaping, a greater port range instead of hand-selected ports, or more variance for hostnames used for DNS tests. Still, we believe that our metrics uncover a variety of techniques. Our work is also limited, as we concentrated on five major Internet products of large Austrian-based providers. However, our methodology is applicable for tests with more providers. Also, some manual work had to be done to evaluate the large number of results. An automated detection of questionable techniques could improve this situation.

4.2 Geographically Decoupled Measurements in Cellular Networks

Cellular networks have become a major access technology to the public phone network and the Internet. In fact, in many parts of the world, it is the only one. As the world continuously grows together, formerly isolated networks need to be considered as much more complex compound systems. In fact, free-roaming agreements such as in the European Union erase borders for end customers and let them perceive cellular service in a unified fashion.

For decades, telecoms (and Internet providers) constituted a natural geographic monopoly of some sort, and so many measurements actually test the providers' infrastructure towards the outside world in general (e.g., firewalls, traffic management) or towards specific other providers (e.g., routing). While digital cellular phone networks have provided roaming capabilities for decades, their prohibitively high pricing made them a sparsely used luxury. Starting in 2007 [161], the EU has been regulating maximum charges on phone calls but not data. In June 2017, EU member states plus Iceland, Liechtenstein, and Norway abolished roaming fees for calls, text, and data under the "roam like at home" doctrine [162]. Since cellular networks in those 31 countries *feel* and behave like the home operator in billing and many other properties, this kick-started a drastic increase in roaming traffic [163].

This development moved data roaming from a mostly theoretical possibility to a daily used service now enabled on most phones. However, since the cellular networks of those 31 countries behave like one towards the user, they should also be measured accordingly. While large-scale Internet measurements became a popular method for a wide range of research questions, this is not true for the mobile sector. We believe this is due to the poor scalability properties of controlled mobile measurement systems based on the requirement of physical SIM cards. In this section, we propose a system for geographically independent cellular network measurements to facilitate large-scale international investigations within minutes. We achieve this by decoupling the mobile identity from the used measurement hardware, i.e., allowing to test roaming effects on a large number of operators without physically move any hardware between member states.

In detail, our contribution is as follows:

- We describe the technique of SIM tunneling to remotely test mobile networks.
- We develop an open, scalable, and controllable measurement framework for mobile Internet connections using our technique of SIM tunneling and deployed measurement probes in multiple European countries.

The remainder of the section is structured as follows: In Section 4.2.1, we describe our methodology, followed by the implementation of the system in Section 4.2.2. Afterwards, we discuss our work in Section 4.2.4.

4.2.1 Methodology

Here, we present our methods of geographic decoupling and address ethical considerations.

Geographic Decoupling of Phone and SIM

A SIM is a smartcard microprocessor issued by a cellular network operator containing cryptographic material and functionality for authentication and authorization of the user's handset. Introduced with Global System for Mobile Communications (GSM), renamed to Universal Subscriber Identity Module (USIM) for Universal Mobile Telecommunications System (UMTS) and LTE, allows the decoupling of handsets, handset manufacturers, network operators, and their customers. The users' identities are instead tied to relocatable modules to be inserted into handsets. Thus, a handset and the SIM are often viewed as an indivisible tandem as one cannot work without the other.

Traditionally, testing mobile connections with SIMs from different operators in different networks (e.g., roaming) includes either (a) physically moving SIMs, (b) physically moving SIMs and their handsets, or (c) replicating each SIM vs. network setup in every territory — thus, scaling more than poorly.

There are multiple ways of geographically decoupling the identity (i.e., SIM) from the actual usage of that identity at a particular network. The baseband of a handset or modem is a radio stack that, at higher layers, communicates a low-bandwidth protocol with the SIM. Walking down the stack, it produces radio messages in the lower layers until it modulates high-bandwidth, low-latency digital radio samples. At the lowest layers, it transforms them into an analog Radio frequency (RF) signal and amplifies it before radiating it over an antenna (and vice-versa for receiving). Thus, geographical displacement of the lower layers comes entailed with much higher engineering overhead than with the higher layers. The high latency tolerance and low-bandwidth communication between the SIM and the radio is the most straightforward point for geographic decoupling.

Thus, we opt for tunneling the SIM card's protocol [164] over the Internet by electrically connecting the modem's SIM socket with General-purpose input/output (GPIO) pins of our measurement platform and simulating the protocol in software. Connecting any SIM with any radio module regardless of geography boosts the automatability of tests across a large number of SIMs and radio networks. SIMs can be both pooled centrally or hosted de-centrally with an infrastructure to dynamically add and remove SIMs.

Ethical Considerations

Ethical considerations are vital to the field of measurements, especially with measurements conducted in live systems. We considered diverse aspects throughout our study, but our focus is threefold: (i) SIM registration laws, (ii) influencing the live network, and (iii) installing our probe – an external device – in a guest network.

In the wake of many states viewing anonymous communication of their citizens as a threat, an increasing number has started requiring the personalization or registration of

SIM cards [165]. As of December 2018, approximately 150 states require some form of SIM card registration [166]. Since some measurements could be viewed as illegitimate use by the cellular network operator, we only used SIM cards registered to ourselves for this paper. Blocked SIM cards would not affect any other user.

The measurement framework operates in real mobile networks. Tests reflect normal user behavior (data access, Short Message Service (SMS) functions, and telephony calls) and should not affect the network or other clients in an unexpected way. However, with SIM tunneling, our SIM cards can change the country in an irregular and unnaturally fast way. This might spark confusion amount the operator systems or trigger fraud control. To address these cases, we manually restrict the minimum duration between country switches to two hours.

Probing, pentesting, but also so benign-seeming activity such as speed measurements could – in some or all cases – violate the operator's terms and conditions. Since we do take precautions not to harm the network and we do not enrich ourselves, we see such testing to be the only way to ascertain robust knowledge of public infrastructure. All used cellular modems are internationally certified for usage by radio regulatory bodies and no attempts were made to circumvent regulatory constraints.

Ultimately, our probes are devices connected to foreign local networks. To facilitate secure deployment in these foreign networks, we ensure that the Internet-facing control traffic is only connecting to the control server via a Virtual private network (VPN). The connection is encrypted and cannot be attributed to the hosts of our measurement stations in a foreign country.

4.2.2 Implementation

In this section, we describe the architecture, implemented techniques, and the deployment of our measurement framework.

Measurement Framework

Our fundamental architecture consists of five components, as illustrated in Figure 4.3:

(i) Probes that execute experiments and measurements on the cellular network via a modem. Furthermore, the probe electrically emulates the SIM card for the modem. A separate Internet connection for management purposes is protected via a VPN.

(ii) SIM providers that connect to multiple genuine operator SIM cards (i.e., concentrate them at one or more places) allow connecting those to any modem via a virtual circuit.

(iii) The actual SIM cards, which contain the secret key material from the operators.

(iv) A VPN connecting all components.

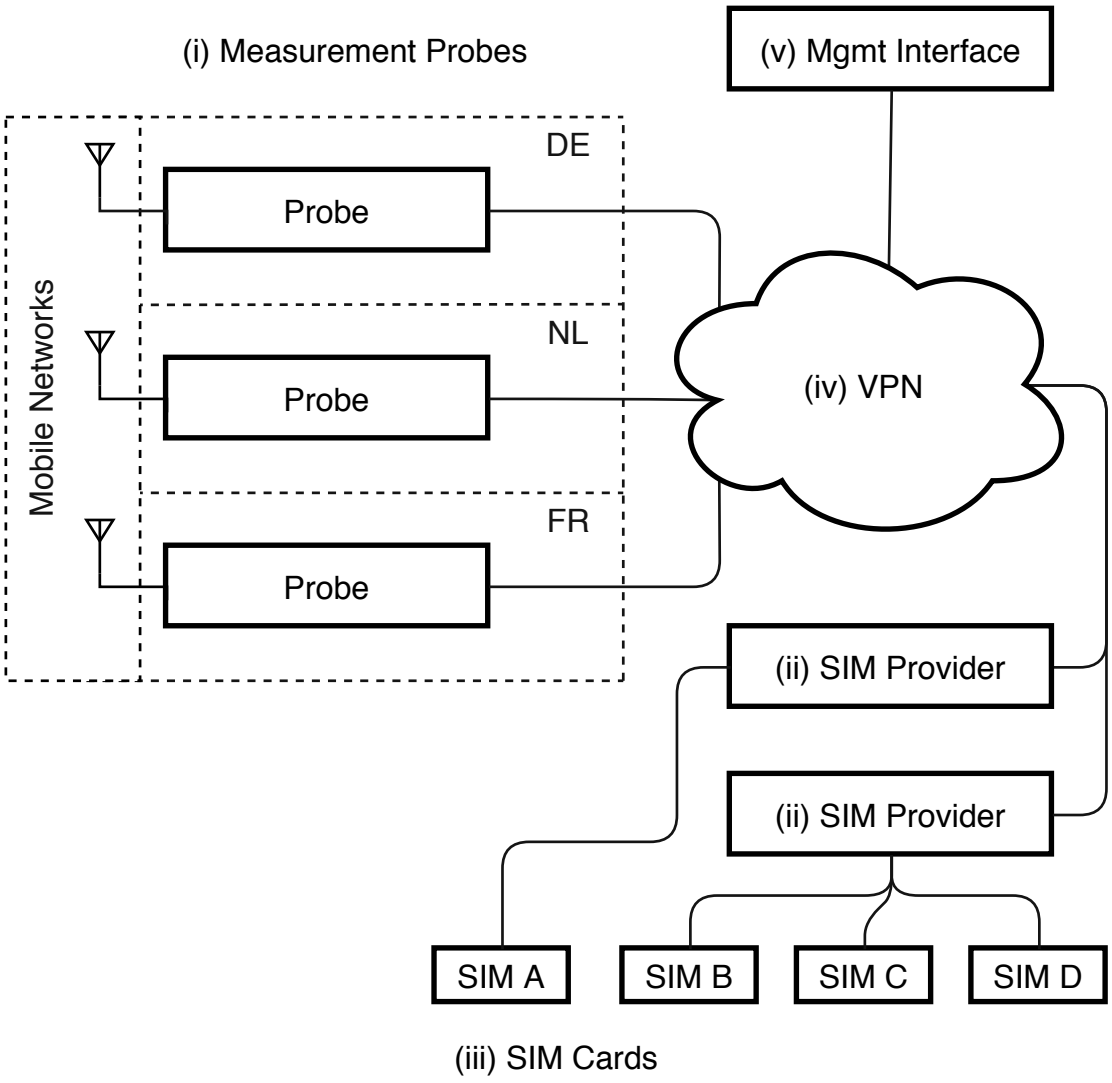


Figure 4.3: Measurement architecture

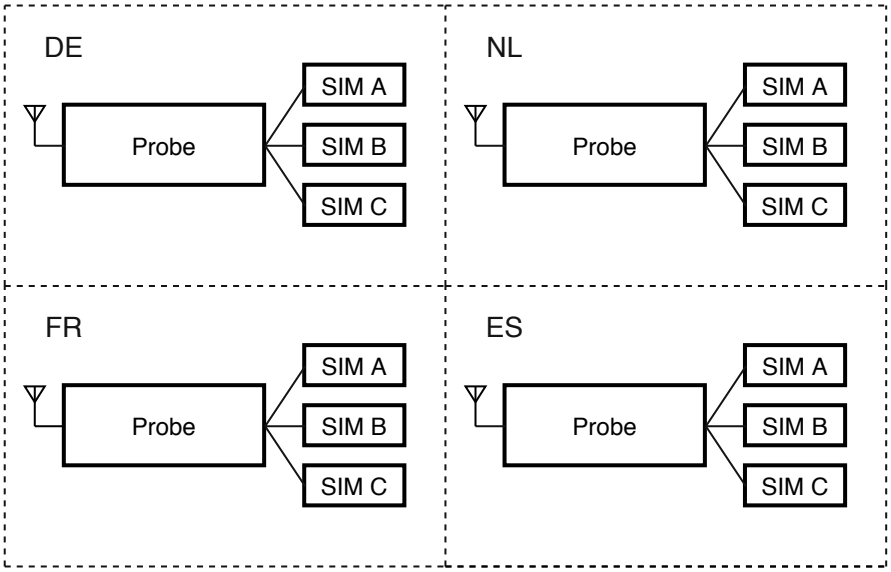


Figure 4.4: Naive architecture with poor scalability properties $\mathcal{O}(|P| \times |S|)$ or $\mathcal{O}(n^2)$, when $n \approx |P| \approx |S|$.

(v) A management interface to dynamically administrate the components and schedule tests.

This architecture allows every probe to use SIM cards attached to SIM providers, independent from the geographical location of a probe and SIM provider. This architecture allows us to use any SIM card at any modem in any country without physically moving it.

In contrast, a more naive architecture as used in previous works requires adding that operator's SIM to every measurement location or transfer them physically (Figure 4.4). SIMs (or actually the provider subscriptions) are the highest cost and maintenance drivers in such measurement systems. The naive approach scales in $\mathcal{O}(|P| \times |S|)$, where $|P|$ is the number of probes, and $|S|$ the number of SIMs. We reduce the cost to linear in regards to the number of subscriptions tested and probes deployed, i.e., it scales in $\mathcal{O}(|P| + |S|)$.

Probes

Measurement probes are responsible for measurement execution. Each probe consists of a single-board computer with at least an Internet uplink, Universal Serial Bus (USB) host support, and a GPIO interface. We chose the *Raspberry Pi 3B+* as a cost-effective platform with more than sufficient computing power for our tasks. The SIM interface at the modem is emulated via a Universal asynchronous receiver-transmitter (UART).

For user-friendly deployment, we packaged the probe with all modules and antennas into a single case with just two connectors: an RJ45 Ethernet and a power connector. An overview of the design is presented in Figure 4.5. Our design is driven by cost-efficiency.

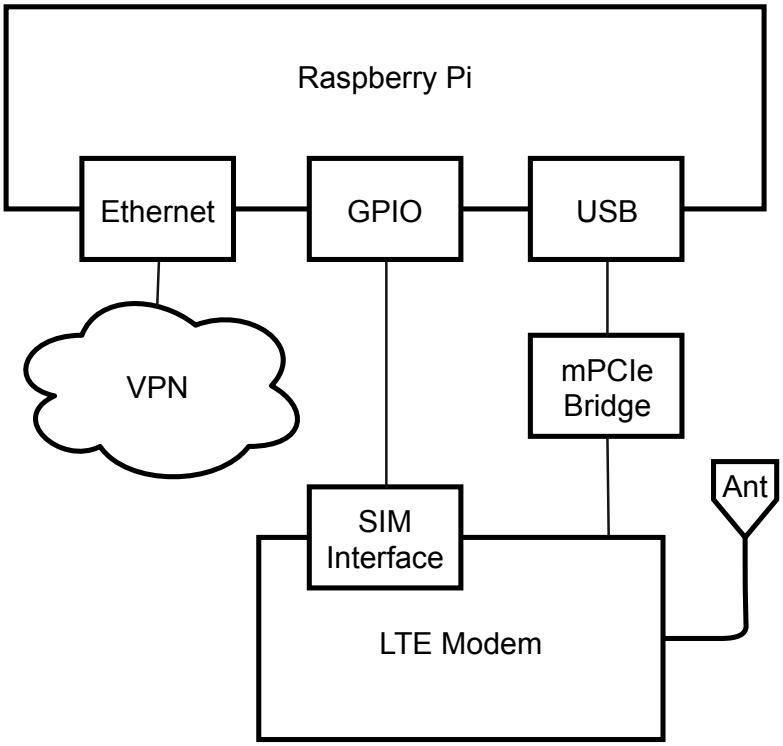


Figure 4.5: Probe architecture design

SIM Providers

SIM providers are responsible for connecting the actual SIM cards to the measurement framework. Dedicated hardware – so-called SIM banks – to connect a multitude of different SIM cards exists and is often used for VoIP gateways. These devices are often expensive, proprietary, and do not perfectly reflect our use case. Instead, we decided on the cost-efficient approach of connecting SIM cards with individual SIM readers. These devices are cheap, readily available, and connect via USB. We were able to successfully test our SIM provider. The framework allows us to add additional distributed SIM cards from other sources, although we currently serve all SIM cards from one SIM provider. With this distributed approach, we facilitate testing a multitude of mobile contracts, as one could temporarily add a SIM card and test it on a large scale. A schematic of a SIM provider is shown in Figure 4.6.

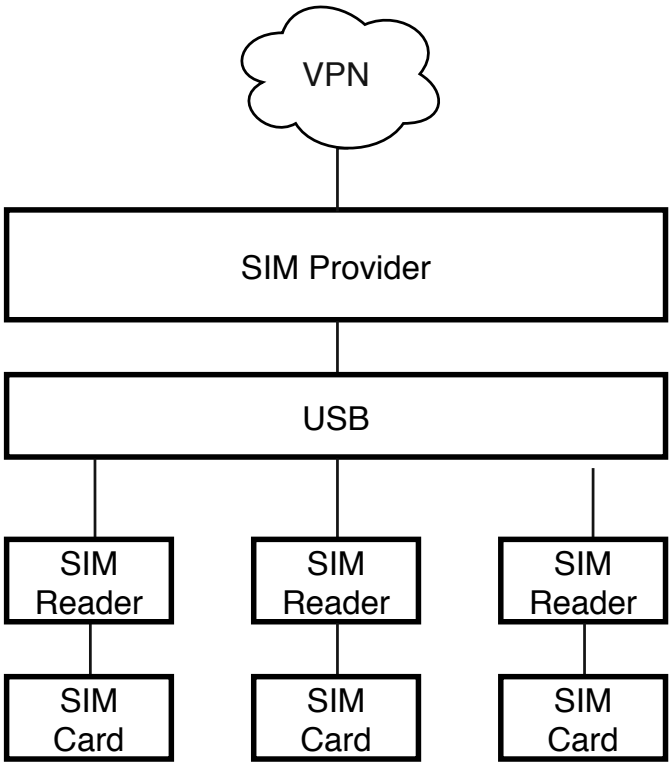


Figure 4.6: SIM provider schematic design

Deployment

We currently deployed probes in six European countries: Austria, France, Germany, the Netherlands, Spain, and the United Kingdom. With these six countries, we already cover the mobile networks of 290 million people. Because our framework is extensible, new hosts could easily extend the set of countries. We argue that for most experiments, one probe per country is sufficient because our test metrics probe network properties and not the properties of individual base stations.

4.2.3 Planned use cases and Experiments

The versatile measurement framework can be used for a variety of different use cases. We plan to use it for:

- IPv4/IPv6 deployment surveys.
- Quality of Experience (QoE) measurements.
- Geoblocking configurations.
- Roaming traffic speed discrimination.

Figure 4.7: Coverage (2020-06)

- OTT and Private branch exchange (PBX) fraud detection [167].
- Free Riding, Zero-Rating, and Fare Dodging detection.

In the following, we describe the first experiments in two use cases: (i) isolated measurements and (ii) scalable roaming measurements.

Isolated Measurements

Unlike app-based measurements, our architecture offers tight control over (background) network traffic and network events. This includes the control and configuration of the TCP stack, modem, and network properties (e.g., selection of cell), and network events (e.g., triggering handovers). Apart from basic network configuration (e.g., access technology), many modem modules have basic functionality for network surveys (e.g., report neighboring cells, signal quality) and cell fixing (e.g., choose a specific cell). Some of those events can trigger traffic outside the user's control. For example, a new data

connection will trigger an IP configuration (e.g., via Dynamic Host Configuration Protocol (DHCP)).

To test how the network's metering handles signaling and setup traffic outside the user's control, we connect and disconnect a hundred times to the Internet service. The basic measurement structure is to try out two measurement settings. First, we measure connections without any user traffic. Second, we measure connections with exactly one Internet Control Message Protocol (ICMP) packet (roughly 200 bytes payload). We do this multiple times. We experimented with a few providers working in different networks, when they provided a detailed, itemized list of calls and data consumption. Operators offering only coarse-grained data consumption statistics would potentially need a much higher number of connnect–disconnect cycles. For each measurement, we connected and disconnected to the network precisely 100 times over several minutes without sending any payload data and disconnected after a few seconds. The second run contained exactly one ICMP packet. After several hours of waiting for the billing system, we retrieved the itemized list of data consumption and compared it against the number of established connections.

For empty connections, no provider billed anything. With exactly one ICMP packet, we see differences in the billing between providers. Two providers showed one single connection with a minimal amount of billed traffic (up to 16 KB). In contrast, one provider showed all connections and billed it separately. We noticed that the minimum data consumption per connection is as high as 100 KB, which added up to 10 MB for 100 empty connections. These measurements were conducted in May–June 2020.

This experiment shows the capabilities of our measurement framework.

Scalable Roaming Measurements

Mobile Internet providers are increasingly introducing products with a fine-grained selection of zero-rated services. For instance, differently billed traffic for certain application service providers, e.g., WhatsApp, or other messaging providers. Previous work analyzed traffic differentiation (see Section 2.2) as well as the implications of international data roaming [168]. However, with an increased number of roaming connections, the questions arise if roamed zero-rated traffic is handled differently. We hypothesize that the roamed traffic classification is identical due to home routing, but differences in metering exist. We have different indications that strengthen our assumption; (i) the providers' general terms and conditions are often vague on this behavior; (ii) on several providers, a plethora of data plans for different services, apps, and services of a single app exist, e.g., messaging add-ons including chats – but not videotelephony – over the same app; (iii) service endpoints change over time due to technical reasons leading to complex metering systems; (iv) anecdotal reports of this behavior exist.

The basic structure of this measurement is as follows. First, the available free credits are checked. Second, data streams are sent to various endpoints of the zero-rated applications. We repeatedly access this resource until the necessary traffic size is reached. Third, we

wait for the billing systems to update and recheck the available credits. This process is repeated for each SIM card and each tested country.

We experimented with data plans that included zero-rated traffic and tested it domestically and internationally. Surprisingly, one provider behaves differently under roaming conditions. We see that WhatsApp traffic in the home network is zero-rated, while roamed traffic is subtracted from the general data plan allowance. For the other providers, the traffic is consistently zero-rated, respectively billed.

This experiment shows that roaming differentiation of zero-rated traffic exists, and we proposed a promising tool for a future large-scale analysis.

4.2.4 Discussion

SIM card management The implementation proves the advantages of geographically decoupling the cellular network probes from the cellular subscription modules (SIMs). Thereby, the architecture reduces the hardware complexity (and costs) in comparison to previous cellular network measurement frameworks. The measurement probe has to be assembled only once and can be shipped everywhere, with rather uncomplicated tasks of provisioning and maintenance.

However, the majority of maintenance time and money is spent on *SIM card management*. They have to be purchased, registered, and regularly topped up, with multiple different systems and languages. Thus, they are the biggest challenge to scale up our framework. Many operators do not ship their SIMs internationally, so we need to build a network of helpers to either forward SIM cards or host them. Specifically, because our architecture supports *SIM Provider* from any computer and SIM reader, we plan to facilitate crowdsourcing, i.e., with easy-to-use docker containers and proper control mechanisms.

The rising trend of embedded-SIMs (eSIMs) promises another simplification. ESIMs profiles can be loaded via an app or a QR code onto a phone without a physical artifact, e.g., from online stores. Technically, an eSIM controller simulates the SIM protocol based on the profile. In this regard, eSIMs perfectly fit into our architecture.

Traffic and roaming metering As the cellular operators try to transition from a mere bit-pushing utility to a service provider, they think they need to market Internet (or app) services. To distinguish those high-level services, they need to classify low-level traffic. What started with internal zero-rating to allow a user to manage their (prepaid) accounts free of charge, evolved into a full-blown multi-class classification endeavor. While the marketing departments push out new “services”, the technical departments seem to struggle to keep up the pace. The mapping of services to traffic is not trivial in times of on-demand cloud scalability, virtual servers, and the proliferation of strong cryptography. Having to provide (and meter) the same services over other networks (roaming) under the regulatory policies does not simplify operations either.

In this area of interplay, operator technicians willingly, by regulatory or technical limitations, choose to lower the traffic separation precision and accept to over-provision in the case of doubt. Our framework is – among other use-cases – uniquely qualified to systematically and internationally survey these borderlines. This low precision of traffic classification also opens up surprising possibilities for fare-dodging by bypassing the traffic metering on subscription- and prepaid-based contracts.

Furthermore, the doctrine of "*roam like at home*" opens up an orthogonal problem dimension. Depending on operator and traffic class, it extends to zero-rated traffic in some cases but not in others. We provide a robust platform to uncover such subtle and surprising billing differences.

Framework The SIM provider offers improvement opportunities based on selective caching and locally terminating SIM traffic. Certain known — but in our case unimportant or always identical Application Protocol Data Unit (APDU) requests — should be locally emulated. This includes many local settings, such as the phone book or the list of last calls. Other well-known data in the SIM's file system could be read once and transferred upfront. This will speed up initialization and connection times as it eliminates network round trips.

While our framework currently covers six European countries with six probes, we strive to further increase the number of countries by increasing the number of deployed probes. We are going to openly publish the design and specifications of our probe to allow other researchers to participate. We continuously look for organizations that host probes and offer on-the-ground support for SIM acquisitions. Furthermore, we encourage these researchers to propose new measurement metrics and experiments.

CHAPTER 5

Enhancing Metadata-free Communication

Metadata-free communication channels can be employed to counter the privacy threat of potentially malicious middlemen. They also work on top – as an overlay network – of the traditional technology stack. Anonymity systems can minimize or eliminate metadata visibility, and therefore, we address these systems in this chapter. Tor, the most prominent anonymity system, helps millions of users around the globe to protect their privacy, which has made it a valuable target for malicious actors. As a low-latency anonymity system, it is vulnerable to traffic correlation attacks from strong passive adversaries, such as large ASes. Estimations of the risk posed by such attackers and the evaluation of defense strategies are primarily based on simulations and data retrieved from BGP updates. However, this might only provide an incomplete view of the network and thereby influence the results of such analyses. It has already been acknowledged in previous studies that direct path measurements, e.g., with traceroute, could provide valuable information. However, such measurements were thought to be impossible in the past, because they require the placement of measurement nodes in the same ASes as the respective Tor network nodes. In Section 5.1, we re-evaluate this assumption and show how global measurement frameworks can be used to find strong passive adversaries.

5.1 Actively Probing Routes for Tor AS-level Adversaries with RIPE Atlas

This section is an extended version of [10]. Preliminary work on the topic was conducted by David Schmidt in his bachelor thesis [169].

Tor is the most notable anonymity network, used by 2 to 3 million people on a daily basis and advertising up to 400 Gbit/s of bandwidth by utilizing around 6,500 voluntarily operated Tor relays. It provides anonymity by routing traffic via three different Tor nodes. As a low latency network, due to its design, it is not capable of guaranteeing anonymity in the case of a global passive observer. This form of an attacker is explicitly excluded from the threat model, assuming that such a global passive observer does not exist. Although not global, powerful observers exist, potentially threatening the anonymity of Tor users. However, their capabilities are not exactly clear. One reason for this is the theoretical assumption that the underlying Internet hierarchy is flat and evenly distributed. Trivially, this is not the case, as the Internet is shaped in different tiers and various entities with different levels of control, e.g., Internet Exchange Point (IXP) with a high level of control and small ISPs with a low level of control. Also, the Tor network does not utilize the Internet in an evenly distributed manner, as the location of Tor relays depends on various parameters, e.g., economical (the price of bandwidth) or political (censorship, prosecution) reasons. Prior work [106,107,112] has shown that traffic through the Tor network only takes a limited set of routes on the Internet, making the threat of a powerful passive observer far more likely. They point out that few AS-level entities provide a high proportion of the Tor bandwidth, thus making them powerful entities for traffic correlation attacks. These studies rely on BGP updates and a prediction of routes taken. While they also describe that a traceroute-based approach could potentially yield better results, they also argue that it would not be feasible for measuring AS-level adversaries in the Tor network because of the need to have measurement nodes placed on the same ASes as Tor nodes and destinations. However, with the introduction of the RIPE Atlas network [88], this assumption can no longer be taken for granted. The RIPE Atlas network is a global measurement network, which can be used by researchers to measure Internet connectivity and reachability. It has already been used for several studies [89] concerning network routing [170] as well as censorship measurements [92].

Our work presents a novel method of measuring the routes that traffic takes from and to the Tor network by utilizing active network probing, in contrast to estimations via BGP updates. We do this by utilizing probes that are placed in ASes also in use by Tor relays, Tor users, or Tor connection recipients. For this purpose, we utilize the RIPE Atlas network, which consists of more than 10,000 globally distributed probes connected to many different ASes. To measure the routes a packet takes from and to the Tor network, we execute traceroute commands on these probes and collect information on the ASes observed on the respective paths. With this method, we gather data to create better predictions of powerful adversaries existing on the Internet and thus to improve the anonymity of Tor users.

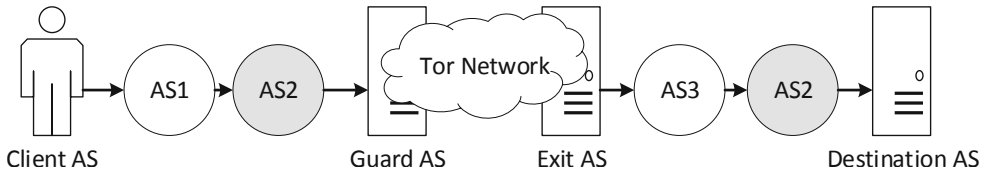


Figure 5.1: AS2 in a possible position for a traffic correlation attack

More specifically, the contributions of this chapter are as follows:

Active Measurements of AS Interconnections: Using *traceroute*-based measurements, we estimate the capabilities of AS-level adversaries and show the influence of only a few ASes on a large amount of traffic.

Open-Source Active Measurement Tool: To improve the evaluation of future attacks and defenses against Tor, we provide an open-source framework to perform active measurements to acquire routing information for Tor nodes.

5.1.1 Active Acquisition of Routing Information

In the following section, we describe a novel method to measure strong AS-level observers, which are in a good position to conduct correlation attacks. As an overlay network, Tor depends on the underlying structure of the Internet. While often a flat hierarchy is assumed, it is clear that this is not the case. We can model the structure of the Internet by looking at ASes identified by a unique Autonomous System Number (ASN). One AS can be seen as an administrative entity that is responsible for a defined routing policy. Some AS are large and include a lot of Tor users, destinations, or relays; others do not contain users and destinations but are used for routing Tor traffic through the Internet, and others are not important for Tor routing at all. Thus, some entities can observe more traffic than others. With our measurements, we find a way to quantify which entities are in a stronger position. Figure 5.1 illustrates the basic idea of a standard traffic correlation attack, where one adversary (AS2) is placed on the incoming route to Tor as well as on the outgoing route to the destination. Sun et al. [111] showed that it is also possible to correlate reverse-path traffic. Other work already quantified strong adversaries with the help of BGP route updates. In contrast, we develop a method that utilizes the RIPE Atlas framework to actively acquire routing information.

Relay AS Diversity

As shown in Table 5.1, the Tor network currently consists of 6,509 relays (January 5th, 2020). Only relays with the *Guard* flag (stable and reliable relays after a ramp-up phase [171]) are used as entry relay. Only relays configured to allow exiting traffic are

	Relays	Diff. AS	BW (Gbit/s)
All Relays	6,509	1,104	418.07
Exit Relays	1,000	275	112.90
Guard Relays	2,415	470	254.61

Table 5.1: Tor Relay overview

(a)  (b) 

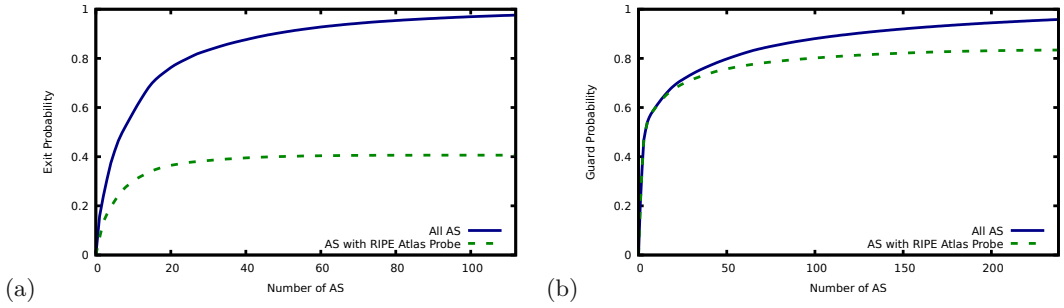
Figure 5.2: (a) Worldwide RIPE Atlas Coverage[1](b) Visualization of Tor Relays[2]

potential exit relays in a Tor circuit. Because of the more stringent requirements, the number of guard and exit relays (with guard/exit probability > 0) is a lot smaller than 6,509. This also affects the AS diversity, which is the number of different ASes these relays are placed in. Current numbers are shown in Table 5.1. Tor relays are chosen based on their flags and consensus weight. In Figure 5.3, we show the AS diversity relation to guard and exit probability. We see that a small number of AS has a large share of exit (a) and guard (b) probability. Eight ASes have more than 50% exit probability, and 48 ASes have more than 90%. We also see that only four ASes have more than 50% guard probability, and 122 have more than 90%. So although all Tor relays are distributed over more than 1,100 ASes, the majority of entry and exit routing endpoints are placed in a few ASes.

The RIPE Atlas Framework

The RIPE Atlas framework is a highly distributed measurement network consisting of more than 10,000 available probes deployed in over 3,500 different ASes. It allows us to execute various low-level commands, e.g., ping or traceroute, on these probes and further process the results. We will utilize this to execute traceroute commands from RIPE Atlas probes that are deployed in the same ASes as Tor guard or exit relays as well as clients and popular destinations. Figure 5.2 illustrates the global distribution of RIPE Atlas probes and the global distribution of Tor relays. We see that countries with a higher number of Tor relays also run more RIPE Atlas probes.

[1] https://atlas.ripe.net/
[2] https://tormap.void.gr/



(a) (b)

Figure 5.3: Accumulated percentage of (a) exit, and (b) guard probability with the number of ASes

AS	Name	Relays	Gbit/s BW	P_{exit}	P_{guard}
200052	FERAL	54	17.01	.158	.004
208323	APPLIEDPRIVACY	16	7.28	.082	.001
53667	FRANTECH	94	8.78	.048	.011
8972	HOSTEUROPE	23	2.60	.000	.010
63949	LINODE-AP	162	3.71	.001	.008

Table 5.2: AS with Tor relays currently not hosting a RIPE Atlas probe

Figure 5.3 also shows the cumulated guard and exit probability for ASes that contain RIPE Atlas probes. From 275 ASes that contain exit relays, only 112 also contain a probe (419 relays out of 1,000). Still, that makes approx. 41% of the total exit probability (35% with only 17 ASes). This differs from the cumulated guard probability. From 470 ASes that contain 2,415 relays, 238 ASes (with 1,848 relays) also include a RIPE Atlas probe, which represent guard relays with a sum of 83% guard probability (80% with 98 ASes). Especially for exit relays, these numbers could be drastically increased if only a few, exit-focused ASes would also host RIPE Atlas probes. Table 5.2 identifies ASes that are currently not hosting any RIPE probes. By adding only 5 probes, we could measure ASes with 76% exit probability in total, and 10 probes would gain up to 87% probability in total.

Active *traceroute* Probing with RIPE Atlas

As illustrated in Figure 5.4, we perform *traceroute* measurements to identify routes taken for four different directions: (1) all client ASes to all guard ASes, (2) exit ASes with probes installed to the destination ASes, (3) destination ASes to all exit ASes, and (4) guard ASes with probes installed to the client ASes. With these measurements, we do not cover all possible routes since not all ASes have probes installed. In the different directions we measure (1) 100% (2) ∼40% (3) 100% (4) ∼83% in terms of route probability.

In detail, this process works as follows:

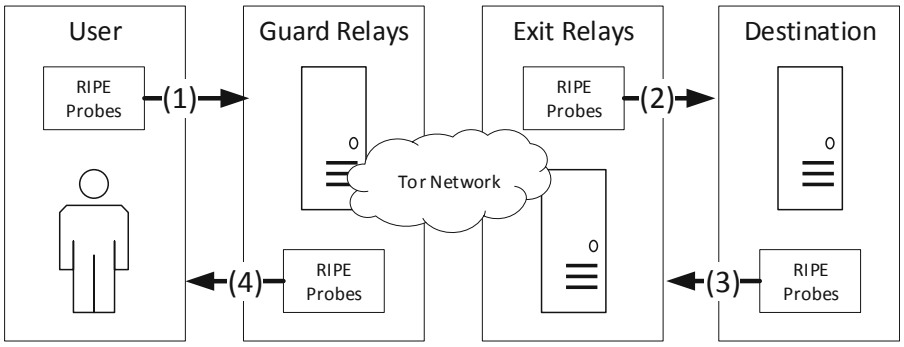


Figure 5.4: Four different directions of active RIPE Atlas *traceroute* scans

1. Create the following sets:
   i. AS_{client} ... ASes of the clients
   ii. AS_{guard} ... all ASes with guard relays
   iii. $AS_{guard+probe}$ ... all ASes with guard relays and RIPE Atlas probes
   iv. AS_{exit} ... all ASes with exit relays
   v. $AS_{exit+probe}$ ... all ASes with exit relays and RIPE Atlas probes
   vi. $AS_{destination}$ ... ASes of the destinations
2. Generate ICMP traceroute measurement definitions for the following directions:
   (1) $AS_{client} \xrightarrow{traceroute} AS_{guard}$
   (2) $AS_{exit+probe} \xrightarrow{traceroute} AS_{destination}$
   (3) $AS_{destination} \xrightarrow{traceroute} AS_{exit}$
   (4) $AS_{guard+probe} \xrightarrow{traceroute} AS_{client}$
3. Execute the *traceroute* with the RIPE Atlas measurement API. (`"protocol": "ICMP", "response_timeout": 20000, "packets": 1`). Every RIPE Atlas measurement is charged with credits, obtained by hosting RIPE Atlas probes. For the current deployment, that estimates to $20 \cdot 1230 = 24600$ credits for one client and one destination.
4. Process all results and look up the corresponding AS from the *ip2asn* database.
5. For every *traceroute*, mark all included ASes with the probability of that path being chosen, i.e., the corresponding guard/ exit probability.

6. Combine the values for directions 1 and 4 for the entry side, and 2 and 3 for the exit side, s.t., if an AS appears on either the forward or the reverse path it is assigned with the probability of that path being chosen. For multiple destinations, all traceroutes are combined.

7. Point out the top ASes, that appear on entry and exit side by looking at $P_{guard} \cap P_{exit}$.

Origin and Destination AS

The sets of guard and exit relays can be derived by combining the Tor consensus with the RIPE Atlas probe overview. However, a client set and a destination set have to be chosen to conduct a measurement. A single client and a single destination are easily scannable, but it doesn't give us a full picture. However, executing traceroutes for all possible client and destination ASes is not feasible. Thus, we have to choose client and destination AS sets for our measurements. In 2008, Edman and Syverson [107] captured traffic from Tor relays to determine top ASes. We are choosing a different approach using popular destinations and large client ASes. For the client set, we choose different countries and pick the 10 ASes containing the most RIPE Atlas probes. Then, we pick one probe thereof. E.g., Germany has 1,485 probes installed, in 343 different ASes, and we pick the ASes with most probes installed[3]. For the US we do the same[4]. For the most common destinations, we derive a list of top destinations from the Tranco [172] top sites list[5]. We take the 100 most popular domains, resolve the domain, and match the corresponding ASes. From the 100 top sites in 44 different ASes, ten ASes also have a RIPE Atlas probe installed[6] and will be used as our destination ASes.

Data Sources

To facilitate reproducibility and encourage openness, all used data files are publicly available at the project website[7]. In particular, our work relies on following data sources:

1. The Tor consensus that contains all Tor relays with their IP address, associated flags (particularly *Guard* and *Exit*), advertised bandwidth and guard and exit probability. We collect this information via the Tor network status protocol *onionoo*[8].

2. Statistical data about the *RIPE Atlas probes*[9]. We use different data (e.g., id, number and AS of the probes) to find all probes connected to the same ASes as guard and exit relays.

[3]Client ASes Germany: 3320, 6830, 31334, 8881, 3209, 6805, 553, 680, 8422, 9145
[4]Client ASes USA: 7922, 701, 7018, 209, 20115, 22773, 5650, 20001, 10796, 11427
[5]Available at https://tranco-list.eu/list/YL6G
[6]Destination ASes: 3, 15169, 4837, 24940, 36351, 14618, 16509, 14907, 3356, 794
[7]Project website: `https://github.com/sbaresearch/ripe-tor`
[8]onionoo: `https://metrics.torproject.org/onionoo.html`
[9]probes: `https://atlas.ripe.net/probes/`

3. Freely accessible *ip2asn*[10] databases to match IP addresses with the corresponding ASN.

4. Active RIPE Atlas *traceroute* results[11]. The measurements used are accessible on the project's website.

5.1.2 Evaluation

In the following section, we evaluate our traceroute scans and show results. We start with an evaluation of a basic scan. Then, we present a larger measurement with multiple clients and destinations. We assess both directions on the guard and the exit side separately and also look at the combined results.

Measurement with a Single Client and a Single Destination

As an illustration of the capabilities of our methodology, we evaluate the results of measurements with one fixed client AS and one fixed destination AS. Therefore, we choose the AS of our research center as $AS_{client} = \{AS1764\}$, and the AS of one mirror of the `torproject.org` website as $AS_{destination} = \{AS24940\}$. We choose RIPE Atlas probes deployed in these ASes (id: 26895, 50609). We then execute 1,240 traceroute commands as defined in Section 5.1.1. Thereof, 269 only contain the client and destination AS, while 971 contain additional ASes on the path. In Table 5.3, we show various results. As expected, the client and destination AS (Hetzner, Nextlayer) are found on all traceroutes. ASes with a high guard or exit probability (Feral, Applied Privacy, OVH) also have a great share, although they are not intermediary and are only found on the single traceroute to/from their AS. Large transit ASes, that appear on many routes are more interesting. In our measurement, we identified AS6939, AS47147, AS1200, and AS174 to be in a powerful position, as they appear on many routes and gain probability of up to 18%. Table 5.4 shows that for this single measurement only few ASes have a probability higher than 1% to appear on both sides.

As described in Section 5.1.1, not all ASes have RIPE Atlas probes installed, $AS_{exit+probes} \rightarrow AS_{destination}$ only represents around 38% of total exit probability and $AS_{guard+probes} \rightarrow AS_{client}$ only represents around 83% of total guard probability. This means real values are estimated to be even higher.

Measurements with Multiple Clients and Multiple Destinations

We conducted the scans on the guard side and exit side separately and afterward combined the results. We conducted 15,160 successful traceroutes for the 10 entry side ASes originating in the US and Germany. On the destination side, we gathered 4,270 successful traceroute results. The scans were performed around 31.12.2019.

[10] ip2asn: `https://iptoasn.com/`
[11] measurements: `https://atlas.ripe.net/measurements`

AS	Name	Dir.	P	P_{relays}	P_{routes}	Routes
24940	HETZNER-AS	exit	.988	.004	.984	269
200052	FERAL	exit	.161	.161	-	1
6939	HURRICANE	exit	.158	.001	.157	20
47147	AS-ANX	exit	.116	-	.116	4
1200	AMS-IX1	exit	.068	-	.068	17
1764	NEXTLAYER	guard	.992	-	.992	454
24940	HETZNER-AS	guard	.202	.202	-	1
16276	OVH	guard	.152	.152	-	1
1200	AMS-IX1	guard	.180	-	.180	55
174	COGENT-174	guard	.095	.007	.088	87

Table 5.3: Results for a single client and single destination

AS	Name	P_{guard}	P_{exit}	$P_{combined}$
24940	HETZNER-AS	.202	.988	.199
1200	AMS-IX1	.180	.068	.012
16276	OVH	.152	.065	.010

Table 5.4: Combined results for a single destination and a single client

Client to Guard Relays We found ASes that have a high probability to appear on the route to/from guard relays. Figure 5.7 shows the probability of different ASes to be on a route to/from a guard relay in the US and Germany. The different data points represent the different originating ASes, and the line the min and max values. We identify ASes in good positions for both countries. AS3356 (LEVEL 3) will be traversed with a high probability for all originating ASes and also has a high probability for the set of German probes. We identify AS1200, AS1273, and AS6830 only in German ASes. AS1299 (TELIANET), AS2914 (NTT-COMMUNICATIONS-2), AS174 (COGENT) and AS9002 (RETN) are strong for both client sets.

Destination Results For the destination set, we use probes of ten different ASes derived from the Tranco Top pages list, as explained in Section 5.1.1. We identified all ASes that were located on the routes for every destination AS. We then combined these values to represent the possibility of a client connecting to all destinations. Table 5.5 shows all ASes that have a probability of over 20%. Figure 5.6 additionally shows the data points for every single destination AS. We excluded all destination ASes, because they appear with certainty and excluded ASes that only appear because exit relays are hosted (AS200052, AS208323 - Applied Privacy).

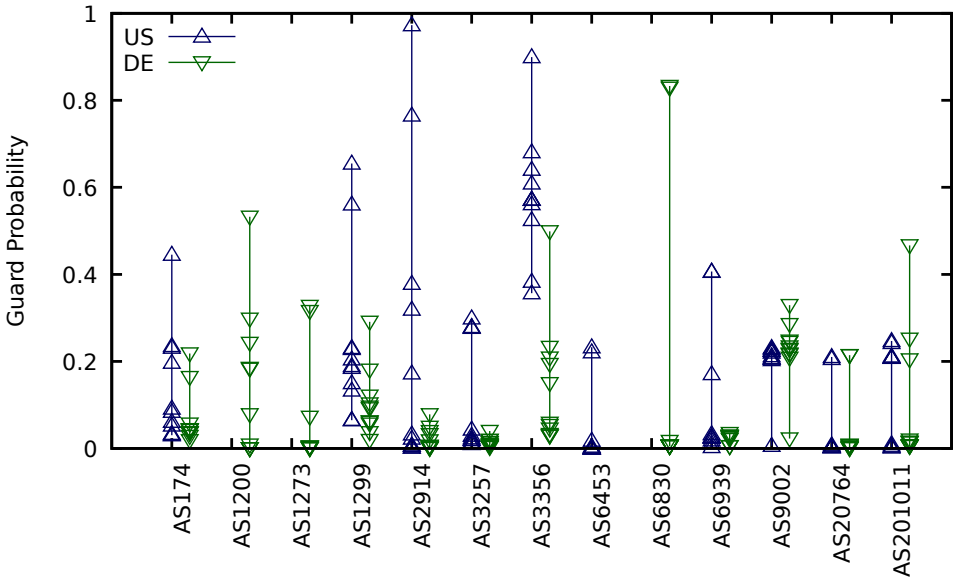


Figure 5.5: Summarized guard probability for ASes that appear on a route from the originating client AS to all guard ASes

AS	Name	P
6939	HURRICANE	0.808
6461	ZAYO-6461	0.510
174	COGENT-174	0.415
1299	TELIANET	0.377
1200	AMS-IX1	0.370
2914	NTT-COMMUN	0.362
10578	GIGAPOP-NE	0.359
3257	GTT-BACKBO	0.290

Table 5.5: Results on the exit side, with a summarized exit probability over 20%

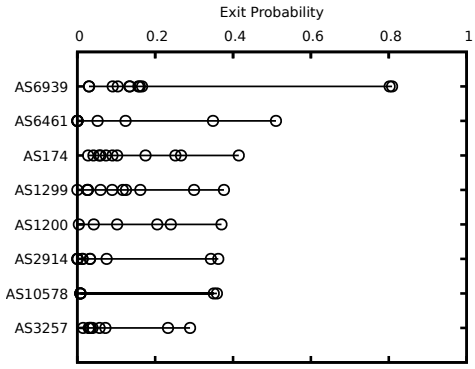


Figure 5.6: Summarized probability with single data points representing the different destination ASes

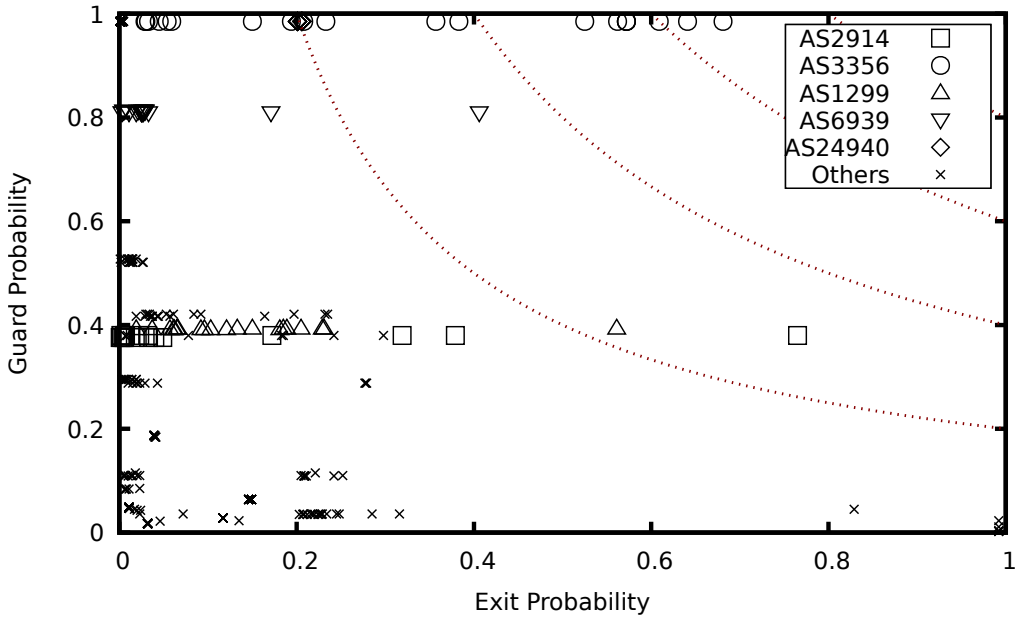


Figure 5.7: Combined probability of ASes appearing on the client and destination path

Combined Results Combining all results, we identify ASes that have a high probability to be on the guard side as well as on the exit side. We investigate combinations of single client ASes with all ASes on the exit side, because users connect from one client AS to different destinations. In Figure 5.7, we can identify strong ASes for our measurement setup. AS3356 (LEVEL3) has a combined value of up to 67.1% ($P_{guard} = .681 \cdot P_{exit} = .985$) for the client AS7018. Other notable ASes are AS6939, AS1299, and AS2914 with combined values $> 20\%$.

5.1.3 Discussion

We presented a methodology to utilize the RIPE Atlas network to gather valuable routing data from and to Tor relays. Related work already quantified AS-level adversaries' capabilities for traffic correlation attacks. Thus, our work does not provide any surprising insights. However, our methods can be used to refine existing models with timely and actively gathered routing data. While the result set of this work is rather limited, with only 16,500 executed traceroute commands and a small number of probes utilized, the methodology is highly scalable. For future work, we plan to scale up the number of measurements performed in various ways. First, we want to enlarge the measured client and destination sets. Second, we think about reoccurring scans in contrast to *oneoff* measurements conducted in this work. Last, a more fine-grained measurement, using probes in the same IP subnets as the relays, could improve the results. We publish our source code openly available as free software. This enables other entities, such as large relay operators, to also perform measurements. All measurement results gathered with RIPE Atlas are also openly available and could include valuable results for the Tor network. We argue that large relay operators should deploy RIPE Atlas probes in their networks, not only to further improve our future results but also to enable other measurements. Only a few more probes would increase the coverage significantly. In Section 5.1.1, we identified the largest relay operators (AS-wise) without RIPE Atlas probes. The evaluation illustrates the possibilities of our methodology. However, it is limited in various ways: We currently do not consider various factors that are important to accurately quantify the threat of AS-level observers. This includes user behavior, Tor circuit creation algorithms, and others. Hence, a combination of our data acquisition method with other simulations is necessary to correctly quantify the traffic correlation threat.

Finally, we argue for increased AS diversity in the Tor network. Even with simple measurements, we see that the distribution of Tor relays is skewed. We hope that our measurements can improve the informed decision on how this diversity should be achieved.

CHAPTER 6

Conclusion

In this thesis, we provided several improvements for measurements of privacy-enhancing technologies.

We summarized and extended the results of a large-scale study, including more than 10 billion TLS handshakes conducted, and discovered multiple flaws that prevent the use of TLS as an effective countermeasure against passive attackers. Overall, the state of TLS in e-mail transmission is, unsurprisingly, worse than compared to HTTPS, as 15–30% of all servers accept weak export grade ciphers, and the majority of certificates are self-signed. On the positive side, we were able to show that RC4 and the use of SSL 2 and SSL 3 for backward compatibility can be considered almost obsolete. According to our calculations, disabling them has hardly any negative impact on the total number of reachable SMTP servers.

We presented existing and new approaches for cipher suite scanning, which is an important tool to evaluate the current status of the TLS ecosystem. Until now, naive approaches were used, which are not optimal in terms of connections, scanning time, or traffic transmitted over the wire. We introduced three new approaches that make use of the TLS protocol specification, common configurations, and existing results. We evaluated the performance gain of these methods and found that we were able to perform scans 3.2 times faster with only 6% of the connections. We implemented a version of the described methods to work with a commonly used tool, simulated them, and then evaluated them in practice by conducting a cipher suite scan for Alexa and the Umbrella top-10k hosts, describing the results and common patterns.

We described the long and challenging process of fully adopting HTTPS. This process is plastered with obstacles, including usability problems. We measured these problems with technical metrics within a qualitative user study. After identifying the problems, we showed multiple approaches to overcome these challenges.

We introduced a new approach to create and validate rules for HTTPS Everywhere. We showed that the currently used approach does not scale and does not use the large number of simple-configured HTTPS deployments. We presented the design of our webservice TLScompare and explained how crowdsource approaches can be used to generate and validate rules. The service was implemented, deployed in a publicly available way, and it collected over 7,500 results for different datasets. We gained first results for existing comparisons of HTTPS Everywhere rules and identified problems of the user's understanding of different terms. We tested newly generated rules and showed that it is possible to consider a rewrite rule to be valid with the use of multiple results per comparison. These rewrite rules were then transformed into an XML representation that is to be submitted to HTTPS Everywhere. We identified problems with the quality of our data and suggested methods of ensuring high data quality with statistical methods. Other approaches of rule generation and additional use cases were considered. The fundamental problem regarding the lack of existing HTTPS deployments was discussed.

With our long-term net neutrality study, we monitored techniques in place from five different Austrian Internet products. This includes the use of middleboxes for SMTP or HTTP and methods used for blocked and non-existing DNS requests. We conducted our measurements in three periods of three months each. We communicated all inconsistencies to the Austrian Regulatory Authority for Broadcasting and Telecommunications (RTR) after each measurement period.

We solved the scalability problem for cellular network measurement and experiment frameworks. Instead of deploying every SIM at every test location, we can dynamically route SIMs to any test location on the network, thus creating a low entry threshold for unique measurement opportunities in a specific country or internationally. For this purpose, we emulate the SIM card protocol via TCP. We demonstrated its usefulness with first experiments and showed that we have a powerful tool for measuring technical properties and operational parameters.

To address Tor traffic correlation attacks through ASes, we presented a novel way to analyze the network routes taken by traffic from and to the Tor network. While previous research relied on the analysis of BGP routing information and simulations, we proposed a new method to utilize the RIPE Atlas framework to measure network routes. We implemented a measurement framework that utilizes the RIPE Atlas probes to perform traceroute commands between clients, servers, and Tor endpoints to collect information on the ASes involved in traffic routing. Next, we utilized the collected information to create a model of paths to locate and quantify strong observers. By leveraging this methodology, we were able to identify a small set of ASes which have a great influence on the total amount of Tor bandwidth. This shows that the collected information is a valuable additional data source when analyzing attacks and defenses based on AS topology.

List of Figures

List of Tables

Acronyms

3DES Tripple Data Encryption Standard. 41

ACME Automatic Certificate Management Environment. 45, 46

ADSL Asymmetric digital subscriber line. 60, 62

AES Advanced Encryption Standard. 10

APDU Application Protocol Data Unit. 77

API application programming interface. 59, 84

ARP Address Resolution Protocol. 47

AS autonomous system. 5, 8, 17, 79–90, 93, 94

ASN Autonomous System Number. 81, 86

BEAST Browser Exploit Against SSL/TLS. 11

BGP Border Gateway Protocol. 5, 17, 79–81

BREACH Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext. 11

CA Certificate Authority. 21, 29, 31, 44, 47

CRIME Compression Ratio Info-leak Made Easy. 11

CSP Content Security Policy. 12

CSR Certificate Signing Request. 44, 45

CSS Cascading Style Sheets. 51

DANE DNS-based Authentication of Named Entities. 30

DH Diffie–Hellman. 41, 42

DHCP Dynamic Host Configuration Protocol. 75

DKIM DomainKeys Identified Mail. 14, 30

DMARC Domain-based Message Authentication, Reporting and Conformance. 14

DNS Domain Name System. 14, 15, 23, 30, 46, 51, 58, 61–63, 65, 66

DNSSEC Domain Name System Security Extensions. 14, 30

DoH DNS over HTTPS. 14

DoT DNS over TLS. 14

DROWN Decrypting RSA with Obsolete and Weakened eNcryption. 11

DSA Digital Signature Algorithm. 12

ECDHE Elliptic-curve Diffie–Hellman Ephemeral. 10, 22

ECDSA Elliptic Curve Digital Signature Algorithm. 11

EFF Electronic Frontier Foundation. 11, 13, 31, 43

eSIM embedded-SIM. 76

ESNI Encrypted Server Name Indication. 3

GCM Galois/Counter mode. 10

GPIO General-purpose input/output. 68, 71

GSM Global System for Mobile Communications. 68

HIT Human Intelligence Task. 55

HMAC Hash-based message authentication code. 10

HPKP HTTP Public Key Pinning. 29, 34

HSTS HTTP Strict Transport Security. 12, 13, 31, 34, 46, 47, 49, 56

HTML HyperText Markup Language. 49

HTTP Hypertext Transfer Protocol. 5, 13, 18, 23, 43, 45, 48–51, 61–63, 65

HTTPS Hypertext Transfer Protocol Secure. 4, 5, 9–14, 18–20, 23, 28, 29, 34–38, 43–56, 95

IANA Internet Assigned Numbers Authority. 10, 13

ICMP Internet Control Message Protocol. 75, 84

ICSI International Computer Science Institute. 31

IMAP Internet Message Access Protocol. 13, 25, 93

IP Internet Protocol. 3, 4, 11, 12, 17, 21–24, 28, 32, 38, 42, 52, 56, 60, 63, 73, 75, 85, 86, 90

ISO International Organization for Standardization. 61

ISP Internet Service Provider. 3, 5, 8, 14, 15, 20, 23, 47, 57–60, 62, 65, 80

IXP Internet Exchange Point. 80

JS JavaScript. 51

JSON JavaScript Object Notation. 59, 60

LAN local area network. 60

LTE Long-Term Evolution. 60, 62, 68

MD5 Message-digest algorithm 5. 42

MITM man-in-the-middle. 47

MTA Mail Transfer Agent. 31

MTA-STS SMTP MTA Strict Transport Security. 31

MX mail exchanger. 22

OONI Open Observatory for Network Interference. 15, 61, 63

OSI Open Systems Interconnection. 61

OTT Over-the-top. 15, 74

PBX Private branch exchange. 74

PCAP packet capture. 60

PET Privacy-enhancing technology. 1, 2, 6, 9

PGP Pretty Good Privacy. 20

PKI Public Key Infrastructure. 11

PKIX Public Key Infrastructure Exchange. 29

POODLE Padding Oracle On Downgraded Legacy Encryption. 11

POP3 Post Office Protocol 3. 13, 25, 61, 65, 93

QoE Quality of Experience. 73

REST Representational state transfer. 52, 59

RF Radio frequency. 68

RFC Request for Comments. 9, 10, 14, 22

RIPE Réseaux IP Européens. 5, 15, 16, 58–60, 80–86, 90, 93, 95

RSA Rivest–Shamir–Adleman. 10, 12, 21, 24, 42

RTR Austrian Regulatory Authority for Broadcasting and Telecommunications. 92

RTT round-trip time. 63

SHA Secure Hash Algorithm. 10

SIM Subscriber Identity Module. 5, 16, 67–69, 71, 72, 76, 77

SMS Short Message Service. 69

SMTP Simple Mail Transfer Protocol. 13, 14, 20, 22, 24–28, 30, 36, 47, 61–63, 65, 93, 95

SMTPS Simple Mail Transfer Protocol Secure. 13

SNI Server Name Indication. 3

SPF Sender Policy Framework. 14, 30

SSL Secure Sockets Layer. 10, 11, 22, 27, 28, 31, 36, 41, 45

TCP Transmission Control Protocol. 13, 17, 21, 22, 24, 36, 37, 42, 61–65, 74, 93

TLS Transport Layer Security. 2–4, 8–14, 16, 19–24, 26–34, 36, 38, 39, 41–43, 45–47, 61, 62, 65

TOFU Trust on first use. 13

TTL time to live. 62, 63

UART Universal asynchronous receiver-transmitter. 71

UCSD University of California, San Diego. 12

UDP User Datagram Protocol. 61, 63

UMTS Universal Mobile Telecommunications System. 68

URL Uniform Resource Locator. 13, 48, 50, 53, 56

USB Universal Serial Bus. 71, 72

USIM Universal Subscriber Identity Module. 68

VoIP Voice over Internet Protocol. 61, 65, 72

VPN Virtual private network. 69

WSGI Web Server Gateway Interface. 52

Bibliography

[1] Facebook's Zuckerberg Says The Age of Privacy Is Over. https://archive.nytimes.com/www.nytimes.com/external/readwriteweb/2010/01/10/10readwriteweb-facebooks-zuckerberg-says-the-age-of-privac-82963.html. Last visit: 2020-11-23.

[2] GW Van Blarkom, JJ Borking, and JGE Olk. *Handbook of Privacy and Privacy-Enhancing Technologies*. Privacy Incorporated Software Agent (PISA) Consortium, The Hague, 2003.

[3] Wilfried Mayer, Aaron Zauner, Martin Schmiedecker, and Markus Huber. No Need for Black Chambers: Testing TLS in the E-mail Ecosystem at Large. In *International Conference on Availability, Reliability and Security*, 2016.

[4] Wilfried Mayer, Aaron Zauner, Martin Schmiedecker, and Markus Huber. No Need for Black Chambers: Testing TLS in the E-mail Ecosystem at Large. *arXiv preprint*, 10 2015.

[5] Wilfried Mayer and Martin Schmiedecker. Turning Active TLS Scanning to Eleven. In *IFIP International Information Security and Privacy Conference*, 2017.

[6] Katharina Krombholz, Wilfried Mayer, Martin Schmiedecker, and Edgar R. Weippl. "I Have No Idea What I'm Doing" - On the Usability of Deploying HTTPS. In *USENIX Security Symposium*. USENIX Association, 8 2017.

[7] Wilfried Mayer, Katharina Krombholz, Martin Schmiedecker, and Edgar R. Weippl. Securing the Internet, One HTTP 200 OK at a Time. *USENIX login, Winter 2017*, 42, 12 2017.

[8] Wilfried Mayer and Martin Schmiedecker. TLScompare: Crowdsourcing Rules for HTTPS Everywhere. In *Workshop on Empirical Research Methods in Information Security (ERMIS)*, 4 2016.

[9] Wilfried Mayer, Thomas Schreiber, and Edgar Weippl. A Framework for Monitoring Net Neutrality. In *International Conference on Availability, Reliability and Security*, ARES 2018, pages 17:1–17:10, New York, NY, USA, 2018. ACM.

[10] Wilfried Mayer, Georg Merzdovnik, and Edgar Weippl. Actively Probing Routes for Tor AS-level Adversaries with RIPE Atlas. In *IFIP International Information Security and Privacy Conference*, 2020.

[11] Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, August 2018.

[12] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008. Updated by RFCs 5746, 5878, 6176.

[13] Ivan Ristic. *Bulletproof SSL and TLS: Understanding and Deploying SSL/TLS and PKI to Secure Servers and Web Applications*. Feisty Duck, 2014.

[14] Jeremy Clark and Paul C van Oorschot. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certificate trust model enhancements. In *IEEE Symposium on Security and Privacy (SP)*, pages 511–525. IEEE, 2013.

[15] T. Dierks and C. Allen. The TLS Protocol Version 1.0. RFC 2246 (Proposed Standard), January 1999. Obsoleted by RFC 4346, updated by RFCs 3546, 5746, 6176.

[16] P. Chown. Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS). RFC 3268 (Proposed Standard), June 2002. Obsoleted by RFC 5246.

[17] IANA. Transport Layer Security (TLS) Parameters. `https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml`. Last visit: 2021-03-15.

[18] Andrey Popov. Prohibiting RC4 Cipher Suites. RFC 7465, February 2015. Updated by RFC 8996.

[19] Y Sheffel, R Holz, and P Saint-Andre. Summarizing Known Attacks on Transport Layer Security (TLS) and Datagram TLS(DTLS). RFC 7457 (Proposed Standard), 2015.

[20] Thai Duong and Juliano Rizzo. Here Come The ⊕ Ninjas. `http://www.hpcc.ecs.soton.ac.uk/dan/talks/bullrun/Beast.pdf`, 2011.

[21] Bodo Möller, Thai Duong, and Krzysztof Kotowicz. This POODLE bites: exploiting the SSL 3.0 fallback. `https://www.openssl.org/~bodo/ssl-poodle.pdf`, 2014.

[22] Juliano Rizzo and Thai Duong. The CRIME Attack. `https://docs.google.com/presentation/d/11eBmGiHbYcHR9gL5nDyZChu_-lCa2GizeuOfaLU2HOU/edit`, 2012. Last visit: 2020-07-28.

[23] Yoel Gluck and Angelo (Angel) Harris, Neal andnd Prado. BREACH: Reviving the CRIME Alttack. http://www.breachattack.com/resources/BREACH-SSL,gonein30seconds.pdf, 2013.

[24] Nadhem J Al Fardan and Kenneth G Paterson. Lucky thirteen: Breaking the TLS and DTLS record protocols. In *IEEE Symposium on Security and Privacy (SP)*, pages 526–540. IEEE, 2013.

[25] Christopher Meyer, Juraj Somorovsky, Eugen Weiss, Jörg Schwenk, Sebastian Schinzel, and Erik Tews. Revisiting SSL/TLS Implementations: New Bleichenbacher Side Channels and Attacks. In *USENIX Security Symposium*, pages 733–748. USENIX Association, 2014.

[26] Matteo Avalle, Alfredo Pironti, and Riccardo Sisto. Formal Verification of Security Protocol Implementations: A Survey. *Formal Aspects of Computing*, 26(1):99–123, 2014.

[27] Benjamin Beurdouche, Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cédric Fournet, Markulf Kohlweiss, Alfredo Pironti, Pierre-Yves Strub, and Jean Karim Zinzindohoue. A Messy State of the Union: Taming the Composite State Machines of TLS. In *IEEE Symposium on Security and Privacy (SP)*. IEEE, 2015.

[28] Chad Brubaker, Suman Jana, Bonnie Ray, Sarfraz Khurshid, and Vitaly Shmatikov. Using Frankencerts for Automated Adversarial Testing of Certificate Validation in SSL/TLS Implementations. In *IEEE Symposium on Security and Privacy (SP)*, pages 114–129. IEEE, 2014.

[29] Karthikeyan Bhargavan, Cédric Fournet, Markulf Kohlweiss, Alfredo Pironti, and P Strub. Implementing TLS with Verified Cryptographic Security. In *IEEE Symposium on Security and Privacy (SP)*, pages 445–459. IEEE, 2013.

[30] David Kaloper-Meršinjak, Hannes Mehnert, Anil Madhavapeddy, and Peter Sewell. Not-quite-so-broken TLS: lessons in re-engineering a security protocol specification and implementation. In *USENIX Security Symposium*. USENIX Association, 2015.

[31] Zakir Durumeric, Frank Li, James Kasten, Johanna Amann, Jethro Beekman, Mathias Payer, Nicolas Weaver, David Adrian, Vern Paxson, Michael Bailey, and J. Alex Halderman. The matter of Heartbleed. In *ACM Internet Measurement Conference (IMC)*, pages 475–488. ACM, November 2014.

[32] Microsoft Security Bulletin MS14-066 - Critical: Vulnerability in Schannel Could Allow Remote Code Execution (2992611). https://docs.microsoft.com/en-us/security-updates/securitybulletins/2014/ms14-066, 2014. Last visit: 2020-07-28.

[33] Nimrod Aviram, Sebastian Schinzel, Juraj Somorovsky, Nadia Heninger, Maik Dankel, Jens Steube, Luke Valenta, David Adrian, J Alex Halderman, Viktor

Dukhovni, et al. DROWN: Breaking TLS Using SSLv2. In *USENIX Security Symposium*, pages 689–706. USENIX Association, 2016.

[34] Applied Crypto Hardening. `https://bettercrypto.org`. Last visit: 2020-07-28.

[35] Y Sheffer, R Holz, and P Saint-Andre. Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS). RFC 7525 (Proposed Standard), 2015.

[36] Homin K Lee, Tal Malkin, and Erich Nahum. Cryptographic Strength of SSL/TLS Servers: Current and Recent Practices. In *ACM Internet Measurement Conference (IMC)*, pages 83–92. ACM, October 2007.

[37] John Heidemann, Yuri Pradkin, Ramesh Govindan, Christos Papadopoulos, Genevieve Bartlett, and Joseph Bannister. Census and Survey of the Visible Internet. In *ACM Internet Measurement Conference (IMC)*, pages 169–182. ACM, October 2008.

[38] Peter Eckersley and Jesse Burns. DEFCON 18 - An Observatory for the SSLiverse. `https://www.eff.org/files/defconssliverse.pdf`, July 2010. Last visit: 2020-07-28.

[39] Ralph Holz, Lothar Braun, Nils Kammenhuber, and Georg Carle. The SSL Landscape – A Thorough Analysis of the X.509 PKI Using Active and Passive Measurements. In *ACM Internet Measurement Conference (IMC)*, pages 427–444. ACM, November 2011.

[40] Bernhard Amann, Matthias Vallentin, Seth Hall, and Robin Sommer. Revisiting SSL: A Large-Scale Study of the Internet's Most Trusted Protocol. Technical Report TR-12-015, ICSI, December 2012.

[41] Bernhard Amann, Robin Sommer, Matthias Vallentin, and Seth Hall. No Attack Necessary: The Surprising Dynamics of SSL Trust Relationships. In *ACM Annual Computer Security Applications Conference (ACSAC)*, pages 179–188. ACM, 2013.

[42] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *USENIX Security Symposium*. USENIX Association, August 2013.

[43] Zakir Durumeric, James Kasten, Michael Bailey, and J Alex Halderman. Analysis of the HTTPS Certificate Ecosystem. In *ACM Internet Measurement Conference (IMC)*, pages 291–304. ACM, October 2013.

[44] Jing Zhang, Zakir Durumeric, Michael Bailey, Mingyan Liu, and Manish Karir. On the Mismanagement and Maliciousness of Networks. In *Symposium on Network and Distributed System Security (NDSS)*. The Internet Society, 2013.

[45] Lin-Shung Huang, Shrikant Adhikarla, Dan Boneh, and Charlie Jackson. An Experimental Study of TLS Forward Secrecy Deployments. *IEEE Internet Computing*, 18(6):43–51, 2014.

[46] Michael Kranch and Joseph Bonneau. Upgrading HTTPS in Mid-Air: An Empirical Study of Strict Transport Security and Key Pinning. In *Symposium on Network and Distributed System Security (NDSS)*. The Internet Society, February 2015.

[47] Michael Weissbacher, Tobias Lauinger, and William Robertson. Why is CSP Failing? Trends and Challenges in CSP Adoption. In *Symposium on Recent Advances in Intrusion Detection (RAID)*, pages 212–233. Springer, 2014.

[48] Nadia Heninger, Zakir Durumeric, Eric Wustrow, and J Alex Halderman. Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices. In *USENIX Security Symposium*, pages 205–220. USENIX Association, August 2012.

[49] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Béguelin, , and Paul Zimmermann. Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice. In *ACM Conference on Computer and Communications Security (CCS)*. ACM, October 2015.

[50] Nadhem J AlFardan, Daniel J Bernstein, Kenneth G Paterson, Bertram Poettering, and Jacob CN Schuldt. On the Security of RC4 in TLS. In *USENIX Security Symposium*, pages 305–320. USENIX Association, August 2013.

[51] Scott Yilek, Eric Rescorla, Hovav Shacham, Brandon Enright, and Stefan Savage. When Private Keys are Public: Results from the 2008 Debian OpenSSL Vulnerability. In *ACM Internet Measurement Conference (IMC)*, pages 15–27. ACM, November 2009.

[52] Arjen Lenstra, James P Hughes, Maxime Augier, Joppe Willem Bos, Thorsten Kleinjung, and Christophe Wachter. Ron was wrong, Whit is right. Technical report, IACR, 2012.

[53] Ralph Holz, Johanna Amann, Olivier Mehani, Matthias Wachs, and Mohamed Ali Kaafar. TLS in the wild: an Internet-wide analysis of TLS-based protocols for electronic communication. In *Symposium on Network and Distributed System Security (NDSS)*. The Internet Society, February 2016.

[54] Qualys SSL Labs. SSL Server Test. `https://www.ssllabs.com/ssltest/`. Last visit: 2017-09-06.

[55] Benjamin Van der Sloot, Johanna Amann, Matthew Bernhard, Zakir Durumeric, Michael Bailey, and J Alex Halderman. Towards a complete view of the certificate ecosystem. In *ACM Internet Measurement Conference (IMC)*, pages 543–549. ACM, 2016.

[56] Zakir Durumeric, David Adrian, Ariana Mirian, Michael Bailey, and J Alex Halderman. A search engine backed by Internet-wide scanning. In *ACM Conference on Computer and Communications Security (CCS)*, pages 542–553. ACM, 2015.

[57] Alberto Dainotti, Alistair King, kc Claffy, Ferdinando Papale, and Antonio Pescapè. Analysis of a "/0" Stealth Scan from a Botnet. In *ACM Internet Measurement Conference (IMC)*, pages 1–14. ACM, 2012.

[58] Zakir Durumeric, Michael Bailey, and J Alex Halderman. An Internet-Wide View of Internet-Wide Scanning. In *USENIX Security Symposium*. USENIX Association, August 2014.

[59] Alexa Internet Inc., Top 1,000,000 sites. `http://s3.amazonaws.com/alexa-static/top-1m.csv.zip`. Last visit: 2020-07-28.

[60] Robert Graham. Masscan: the entire Internet in 3 minutes. `http://blog.erratasec.com/2013/09/masscan-entire-internet-in-3-minutes.html`. Last visit: 2015-06-17.

[61] J. Hodges, C. Jackson, and A. Barth. HTTP Strict Transport Security (HSTS). RFC 6797 (Proposed Standard), November 2012.

[62] David Keeler. Preloading HSTS. Mozilla Security Blog - `https://blog.mozilla.org/security/2012/11/01/preloading-hsts`, 2012. Last visit: 2020-07-28.

[63] HTTPS Everywhere. `https://www.eff.org/de/https-everywhere`. Last visit: 2021-03-15.

[64] Sascha Fahl, Marian Harbach, Thomas Muders, Lars Baumgärtner, Bernd Freisleben, and Matthew Smith. Why Eve and Mallory Love Android: An Analysis of Android SSL (In)Security. In *ACM Conference on Computer and Communications Security (CCS)*, pages 50–61. ACM, 2012.

[65] Martin Georgiev, Subodh Iyengar, Suman Jana, Rishita Anubhai, Dan Boneh, and Vitaly Shmatikov. The Most Dangerous Code in the World: Validating SSL Certificates in Non-Browser Software. In *ACM Conference on Computer and Communications Security (CCS)*, pages 38–49. ACM, 2012.

[66] Sascha Fahl, Marian Harbach, Henning Perl, Markus Koetter, and Matthew Smith. Rethinking SSL Development in an Appified World. In *ACM Conference on Computer and Communications Security (CCS)*, pages 49–60. ACM, 2013.

[67] Boyuan He, Vaibhav Rastogi, Yinzhi Cao, Yan Chen, VN Venkatakrishnan, Runqing Yang, and Zhenrui Zhang. Vetting SSL Usage in Applications with SSLINT. In *IEEE Symposium on Security and Privacy (SP)*. IEEE, 2015.

[68] J. Klensin. Simple Mail Transfer Protocol. RFC 5321 (Draft Standard), October 2008.

[69] P. Hoffman. SMTP Service Extension for Secure SMTP over Transport Layer Security. RFC 3207 (Proposed Standard), February 2002.

[70] J. Myers and M. Rose. Post Office Protocol - Version 3. RFC 1939 (INTERNET STANDARD), May 1996. Updated by RFCs 1957, 2449, 6186.

[71] C. Newman. Using TLS with IMAP, POP3 and ACAP. RFC 2595 (Proposed Standard), June 1999. Updated by RFC 4616.

[72] M. Crispin. INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1. RFC 3501 (Proposed Standard), March 2003. Updated by RFCs 4466, 4469, 4551, 5032, 5182, 5738, 6186, 6858.

[73] R. Gellens and J. Klensin. Message Submission for Mail. RFC 4409 (Draft Standard), April 2006. Obsoleted by RFC 6409.

[74] V. Dukhovni. Opportunistic Security: Some Protection Most of the Time. RFC 7435 (Informational), December 2014.

[75] Zakir Durumeric, David Adrian, Ariana Mirian, James Kasten, Elie Bursztein, Nicolas Lidzborski, Kurt Thomas, Vijay Eranti, Michael Bailey, and J Alex Halderman. Neither snow nor rain nor MITM... an empirical analysis of email delivery security. In *ACM Internet Measurement Conference (IMC)*. ACM, 2015.

[76] Marcel Dischinger, Massimiliano Marcon, Saikat Guha, P Krishna Gummadi, Ratul Mahajan, and Stefan Saroiu. Glasnost: Enabling End Users to Detect Traffic Differentiation. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 405–418. USENIX Association, 2010.

[77] Nicholas Weaver, Christian Kreibich, and Vern Paxson. Redirecting DNS for Ads and Profit. In *USENIX Workshop on Free and Open Communications on the Internet (FOCI)*. USENIX Association, 2011.

[78] Arash Molavi Kakhki, Abbas Razaghpanah, Anke Li, Hyungjoon Koo, Rajesh Golani, David Choffnes, Phillipa Gill, and Alan Mislove. Identifying Traffic Differentiation in Mobile Networks. In *ACM Internet Measurement Conference (IMC)*, pages 239–251. ACM, 2015.

[79] Xing Xu, Yurong Jiang, Tobias Flach, Ethan Katz-Bassett, David Choffnes, and Ramesh Govindan. Investigating Transparent Web Proxies in Cellular Networks. In *International Conference on Passive and Active Network Measurement (PAM)*, pages 262–276. Springer, 2015.

[80] Arash Molavi Kakhki, Fangfan Li, David Choffnes, Ethan Katz-Bassett, and Alan Mislove. BingeOn Under the Microscope: Understanding T-Mobiles Zero-Rating Implementation. In *ACM Workshop on QoE-based Analysis and Management of Data Communication Networks (Internet-QuE)*. ACM, 2016.

[81] Gabi Nakibly, Jaime Schcolnik, and Yossi Rubin. Website-Targeted False Content Injection by Network Operators. In *USENIX Security Symposium*, pages 227–244. USENIX Association, 2016.

[82] Sheharbano Khattak, David Fifield, Sadia Afroz, Mobin Javed, Srikanth Sundaresan, Damon McCoy, Vern Paxson, and Steven J Murdoch. Do You See What I See? Differential Treatment of Anonymous Users. In *Symposium on Network and Distributed System Security (NDSS)*. The Internet Society, February 2016.

[83] Merve Sahin and Aurélien Francillon. Over-The-Top Bypass: Study of a Recent Telephony Fraud. In *ACM Conference on Computer and Communications Security (CCS)*, pages 1106–1117. ACM, 2016.

[84] Fangfan Li, Arian Akhavan Niaki, David Choffnes, Phillipa Gill, and Alan Mislove. A Large-Scale Analysis of Deployed Traffic Differentiation Practices. In *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM)*, pages 130–144. ACM, 2019.

[85] Tobias Flach, Pavlos Papageorge, Andreas Terzis, Luis Pedrosa, Yuchung Cheng, Tayeb Karim, Ethan Katz-Bassett, and Ramesh Govindan. An Internet-Wide Analysis of Traffic Policing. In *Proceedings of the 2016 ACM SIGCOMM Conference (SIGCOMM)*, pages 468–482. ACM, 2016.

[86] Arturo Filasto and Jacob Appelbaum. OONI: Open Observatory of Network Interference. In *USENIX Workshop on Free and Open Connuncations on the Internet (FOCI)*. USENIX Association, 2012.

[87] RIPE Network Coordination Centre. RIPE Atlas. `https://atlas.ripe.net`. Last visit: 2020-05-29.

[88] RIPE NCC Staff. RIPE Atlas: A Global Internet Measurement Network. *Internet Protocol Journal*, 18(3), 2015.

[89] Vaibhav Bajpai, Steffie Jacob Eravuchira, and Jürgen Schönwälder. Lessons Learned From Using the RIPE Atlas Platform for Measurement Research. *SIGCOMM Computer Communication Review*, 45(3):35–42, 2015.

[90] Ruwaifa Anwar, Haseeb Niaz, David Choffnes, Ítalo Cunha, Phillipa Gill, and Ethan Katz-Bassett. Investigating Interdomain Routing Policies in the Wild. In *ACM Internet Measurement Conference (IMC)*, pages 71–77. ACM, 2015.

[91] Ben Jones, Nick Feamster, Vern Paxson, Nicholas Weaver, and Mark Allman. Detecting DNS Root Manipulation. In *International Conference on Passive and Active Network Measurement (PAM)*, pages 276–288. Springer, 2016.

[92] Collin Anderson, Philipp Winter, and Roya. Global Network Interference Detection Over the RIPE Atlas Network. In *USENIX Workshop on Free and Open Connuncations on the Internet (FOCI)*. USENIX Association, August 2014.

[93] Özgü Alay, Andra Lutu, David Ros, Rafael Garcia, Vincenzo Mancuso, Audun Fosselie Hansen, Anna Brunstrom, Marco Ajmone Marsan, and Hakon Lonsethagen. MONROE: Measuring Mobile Broadband Networks in Europe. In *IRTF & ISOC Workshop on Research and Applications of Internet Measurements (RAIM)*. IRTF & ISOC, 2015.

[94] Özgü Alay, Andra Lutu, Rafael García, Miguel Peón-Quirós, Vincenzo Mancuso, Thomas Hirsch, Tobias Dely, Jonas Werme, Kristian Evensen, Audun Hansen, et al. Measuring and Assessing Mobile Broadband Networks with MONROE. In *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–3. IEEE, 2016.

[95] The MONROE Alliance. Measuring Mobile Broadband Networks in Europe. `https://www.monroe-project.eu`. Last visit: 2020-05-29.

[96] Andreas Pfitzmann and Marit Hansen. A terminology for talking about privacy by data minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management. `http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf`, 2010.

[97] George Danezis and Ross Anderson. The Economics of Censorship Resistance. In *Proceedings of Workshop on Economics and Information Security*, 2004.

[98] George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *IEEE Symposium on Security and Privacy (SP)*, pages 2–15. IEEE, 2003.

[99] David Chaum, Farid Javani, Aniket Kate, Anna Krasnova, Joeri de Ruiter, Alan T Sherman, and D Das. cMix: Anonymization by High-Performance Scalable Mixing. In *USENIX Security Symposium*. USENIX Association, 2016.

[100] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-Generation Onion Router. In *USENIX Security Symposium*. USENIX Association, 2004.

[101] Alex Biryukov, Ivan Pustogarov, and Ralf-Philipp Weinmann. Trawling for Tor Hidden Services: Detection, Measurement, Deanonymization. In *IEEE Symposium on Security and Privacy (SP)*, pages 80–94. IEEE, 2013.

[102] Kyle Soska and Nicolas Christin. Measuring the Longitudinal Evolution of the Online Anonymous Marketplace Ecosystem. In *USENIX Security Symposium*, pages 33–48. USENIX Association, 2015.

[103] Nicolas Christin. Traveling the Silk Road: A Measurement Analysis of a Large Anonymous Online Marketplace. In *International Conference on the World Wide Web (WWW)*, pages 213–224. International World Wide Web Conference Committee (IW3C2), ACM, 2013.

[104] Iskander Sanchez-Rola, Davide Balzarotti, and Igor Santos. The Onions Have Eyes: A Comprehensive Structure and Privacy Analysis of Tor Hidden Services. In *International Conference on the World Wide Web (WWW)*, pages 1251–1260. International World Wide Web Conference Committee (IW3C2), ACM, 2017.

[105] OnionScan: Investigating the DarkWeb. `https://onionscan.org`. Last visit: 2021-03-14.

[106] Nick Feamster and Roger Dingledine. Location Diversity in Anonymity Networks. In *ACM Workshop on Privacy in the Electronic Society (WPES)*. ACM, 2004.

[107] Matthew Edman and Paul Syverson. AS-awareness in Tor path selection. In *ACM Conference on Computer and Communications Security (CCS)*. ACM, 2009.

[108] Aaron Johnson, Chris Wacek, Rob Jansen, Micah Sherr, and Paul Syverson. Users Get Routed: Traffic Correlation on Tor by Realistic Adversaries. In *ACM Conference on Computer and Communications Security (CCS)*, pages 337–348. ACM, 2013.

[109] Chris Wacek, Henry Tan, Kevin S Bauer, and Micah Sherr. An Empirical Evaluation of Relay Selection in Tor. In *Symposium on Network and Distributed System Security (NDSS)*. The Internet Society, 2013.

[110] Laurent Vanbever, Oscar Li, Jennifer Rexford, and Prateek Mittal. Anonymity on QuickSand: Using BGP to Compromise Tor. In *ACM Workshop on Hot Topics in Networks (HotNets)*. ACM, 2014.

[111] Yixin Sun, Anne Edmundson, Laurent Vanbever, Oscar Li, Jennifer Rexford, Mung Chiang, and Prateek Mittal. RAPTOR: Routing Attacks on Privacy in Tor. In *USENIX Security Symposium*. USENIX Association, 2015.

[112] Rishab Nithyanand, Oleksii Starov, Adva Zair, Phillipa Gill, and Michael Schapira. Measuring and Mitigating AS-level Adversaries Against Tor. In *Symposium on Network and Distributed System Security (NDSS)*. The Internet Society, 2016.

[113] Vasileios Giotsas, Matthew Luckie, Bradley Huffaker, and kc claffy. Inferring Complex AS Relationships. In *ACM Internet Measurement Conference (IMC)*. ACM, 2014.

[114] Phillipa Gill, Michael Schapira, and Sharon Goldberg. Modeling on quicksand: Dealing with the scarcity of ground truth in interdomain routing data. *ACM SIGCOMM Computer Communication Review*, 42(1):40–46, 2012.

[115] Nicholas Hopper, Eugene Y Vasserman, and Eric Chan-Tin. How Much Anonymity does Network Latency Leak? *ACM Transactions on Information and System Security*, 13(2), 2010.

[116] Prateek Mittal, Ahmed Khurshid, Joshua Juen, Matthew Caesar, and Nikita Borisov. Stealthy Traffic Analysis of Low-Latency Anonymous Communication Using Throughput Fingerprinting. In *ACM Conference on Computer and Communications Security (CCS)*. ACM, 2011.

[117] Milad Nasr, Alireza Bahramali, and Amir Houmansadr. DeepCorr: Strong Flow Correlation Attacks on Tor Using Deep Learning. In *ACM Conference on Computer and Communications Security (CCS)*. ACM, 2018.

[118] Mashael Alsabah and Ian Goldberg. Performance and Security Improvements for Tor: A Survey. *ACM Computing Surveys (CSUR)*, 49(2), 2016.

[119] Masoud Akhoondi, Curtis Yu, and Harsha V Madhyastha. LASTor: A Low-Latency AS-Aware Tor Client. In *IEEE Symposium on Security and Privacy (SP)*. IEEE, 2012.

[120] Yixin Sun, Anne Edmundson, Nick Feamster, Mung Chiang, and Prateek Mittal. Counter-RAPTOR: Safeguarding Tor Against Active Routing Attacks. In *IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017.

[121] Armon Barton and Matthew Wright. DeNASA: Destination-Naive AS-Awareness inAnonymous Communications. *Proceedings on Privacy Enhancing Technologies (PoPETs)*, 2016.

[122] Armon Barton, Matthew Wright, Jiang Ming, and Mohsen Imani. Towards Predicting Efficient and Anonymous Tor Circuits. In *USENIX Security Symposium*. USENIX Association, 2018.

[123] Aaron Johnson, Rob Jansen, Aaron D Jaggard, Joan Feigenbaum, and Paul Syverson. Avoiding The Man on the Wire: Improving Tor's Security with Trust-Aware Path Selection. In *Symposium on Network and Distributed System Security (NDSS)*. The Internet Society, 2017.

[124] Robert Annessi and Martin Schmiedecker. NavigaTor: Finding Faster Paths to Anonymity. In *IEEE European Symposium on Security and Privacy (EuroSP)*, pages 214–226. IEEE, 2016.

[125] Chen Chen, Daniele E Asoni, David Barrera, George Danezis, and Adrain Perrig. HORNET: High-speed Onion Routing at the Network Layer. In *ACM Conference on Computer and Communications Security (CCS)*, pages 1441–1454. ACM, 2015.

[126] Hans Hanley, Yixin Sun, Sameer Wagh, and Prateek Mittal. DPSelect: A Differential Privacy Based Guard Relay Selection Algorithm for Tor. *Proceedings on Privacy Enhancing Technologies (PoPETs)*, 2019.

[127] Gerry Wan, Aaron Johnson, Ryan Wails, Sameer Wagh, and Prateek Mittal. Guard Placement Attacks on Path Selection Algorithms for Tor. *Proceedings on Privacy Enhancing Technologies (PoPETs)*, 2019.

[128] 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 Symposium (PETS)*, pages 304–331. Springer, 2014.

[129] Markus Huber, Martin Mulazzani, and Edgar Weippl. Tor HTTP usage and information leakage. In *IFIP International Conference on Communications and Multimedia Security*, pages 245–255. Springer, 2010.

[130] Wilfried Mayer. Analysis of Internet-wide TLS Usage Beyond HTTPS, 2015. Master Thesis – TU Wien.

[131] Google. Transparency Report - Email encryption in transit. `https://www.google.com/transparencyreport/saferemail/`. Last visit: 2015-09-16.

[132] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.1. RFC 4346 (Proposed Standard), April 2006. Obsoleted by RFC 5246, updated by RFCs 4366, 4680, 4681, 5746, 6176.

[133] Ralph Holz, Thomas Riedmaier, Nils Kammenhuber, and Georg Carle. X. 509 Forensics: Detecting and Localising the SSL/TLS Men-in-the-Middle. In *European Symposium on Research in Computer Security (ESORICS)*, pages 217–234. Springer, 2012.

[134] Christopher Soghoian and Sid Stamm. Certified Lies: Detecting and Defeating Government Interception Attacks against SSL (Short Paper). In *International Conference on Financial Cryptography and Data Security (FC)*, pages 250–259. Springer, 2012.

[135] Chris Evans, Chris Palmer, and R Sleevi. Public key pinning extension for http (hpkp). RFC 7469 (Draft), 2015.

[136] P. Hoffman and J. Schlyter. The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA. RFC 6698 (Proposed Standard), August 2012. Updated by RFC 7218.

[137] Bernhard Amann, Matthias Vallentin, Seth Hall, and Robin Sommer. Extracting Certificates from Live Traffic: A Near Real-Time SSL Notary Service. Technical Report TR-12-014, ICSI, November 2012.

[138] Keith Moore and Chris Newman. Cleartext Considered Obsolete: Use of Transport Layer Security (TLS) for Email Submission and Access. RFC 8314, January 2018.

[139] Daniel Margolis, Mark Risher, Binu Ramakrishnan, Alex Brotman, and Janet Jones. SMTP MTA Strict Transport Security (MTA-STS). RFC 8461, September 2018.

[140] Kathleen Moriarty and Stephen Farrell. Deprecating TLS 1.0 and TLS 1.1. RFC 8996, March 2021.

[141] Loganaden Velvindron and Stephen Farrell. Deprecation of TLS 1.1 for Email Submission and Access. RFC 8997, March 2021.

[142] Mozilla Wiki. Security/Server Side TLS. `https://wiki.mozilla.org/Security/Server_Side_TLS`. Last visit: 2015-09-26.

[143] SSLyze - Fast and full-featured SSL scanner. `https://github.com/nabla-c0d3/sslyze`. Last visit: 2015-09-26.

[144] D. Hubbard. Cisco umbrella 1 million. `https://blog.opendns.com/2016/12/14/cisco-umbrella-1-million/`, 2016. Last visit: 2020-07-28.

[145] Mozilla SSL Configuration Generator. `https://mozilla.github.io/server-side-tls/ssl-config-generator/`. Last visit: 2020-07-28.

[146] Katharina Krombholz-Reindl. Usable Security and Privacy Challenges with Disruptive Technologies, 2016. PhD Thesis – TU Wien.

[147] Adrienne Porter Felt, Richard Barnes, April King, Chris Palmer, Chris Bentzel, and Parisa Tabriz. Measuring HTTPS Adoption on the Web. In *USENIX Security Symposium*, pages 1323–1338. USENIX Association, 2017.

[148] ACME v2 API Endpoint Coming January 2018. `https://letsencrypt.org/2017/06/14/acme-v2-api.html`. Last visit: 2017-09-06.

[149] Wildcard Certificates Coming January 2018. `https://letsencrypt.org/2017/07/06/wildcard-certificates-coming-jan-2018.html`. Last visit: 2017-09-06.

[150] Markus Huber, Martin Mulazzani, and Edgar Weippl. Who on Earth Is "Mr. Cypher": Automated Friend Injection Attacks on Social Networking Sites. In *Security and Privacy – Silver Linings in the Cloud*, pages 80–89. Springer, 2010.

[151] Flask - web development, one drop at a time. `http://flask.pocoo.org/`. Last visit: 2020-07-28.

[152] SQLAlchemy - The database toolkit for python. `http://www.sqlalchemy.org/`. Last visit: 2020-07-28.

[153] Amazon Mechanical Turk. https://www.mturk.com/mturk/welcome. Last visit: 2020-07-28.

[154] Let's Encrypt. https://letsencrypt.org/. Last visit: 2020-07-28.

[155] Thomas Schreiber. Monitoring Net Neutrality of Austrian ISPs, 2016. Master Thesis – TU Wien.

[156] Robin S Lee and Tim Wu. Subsidizing Creativity through Network Design: Zero-Pricing and Net Neutrality. *The Journal of Economic Perspectives*, 23(3):61–76, 2009.

[157] Body of European Regulators for Electronic Communications (BEREC). All you need to know about Net Neutrality rules in the EU. https://berec.europa.eu/eng/netneutrality/introduction/. Last visit: 2020-07-28.

[158] Electronic Frontier Foundation. Net Neutrality. https://www.eff.org/issues/net-neutrality. Last visit: 2020-07-28.

[159] Jay Pil Choi, Doh-Shin Jeon, and Byung-Cheol Kim. Net Neutrality, Business Models, and Internet Interconnection. *American Economic Journal: Microeconomics*, 7(3):104–141, 2015.

[160] Hsing Kenneth Cheng, Subhajyoti Bandyopadhyay, and Hong Guo. The Debate on Net Neutrality: A Policy Perspective. *Information Systems Research*, 22(1):60–82, 2011.

[161] European Parliament and Council of the European Union. Regulation (ec) no 717/2007 on roaming on public mobile telephone networks within the community and amending directive 2002/21/ec. *OJ*, L 171:32–40, 2007-06-29.

[162] European Parliament and Council of the European Union. Regulation (eu) 2017/920 amending regulation (eu) no 531/2012 as regards rules for wholesale roaming markets. *OJ*, L 147:30–47, 2017-06-09.

[163] Colin Blackman and Simon Forge. Roaming: One year after implementation. Technical report, European Parliament - Policy Department for Economic, Scientific and Quality of Life Policies, 2018.

[164] ISO. ISO7816-3:2006: Identification cards — Integrated circuit cards — Part 3: Cards with contacts — Electrical interface and transmission protocols. Standard, International Organization for Standardization, Geneva, CH, November 2006.

[165] Yiannis Theodorou. The Mandatory Registration of Prepaid SIM Card Users. Technical report, GSM Association, 2013.

[166] Yiannis Theodorou, Ken Okong'o, and Erdoo Yongo. Digital Identity: Access to Mobile Services and Proof of Identity 2019: Assessing the impact on digital and financial inclusion. Technical report, GSM Association, 2019.

[167] Merve Sahin, Aurélien Francillon, Payas Gupta, and Mustaque Ahamad. SoK: Fraud in Telephony Networks. In *IEEE European Symposium on Security and Privacy (EuroSP)*. IEEE, 2017.

[168] Anna Maria Mandalari, Andra Lutu, Ana Custura, Ali Safari Khatouni, Özgü Alay, Marcelo Bagnulo, Vaibhav Bajpai, Anna Brunstrom, Jörg Ott, Marco Mellia, et al. Experience: Implications of Roaming in Europe. In *ACM International Conference on Mobile Computing and Networking (MobiCom)*, pages 179–189. ACM, 2018.

[169] David Schmidt. RipeTor: Measuring Correlation Attacks Against Tor, 2018. Bachelor Thesis – TU Wien.

[170] Rodérick Fanou, Pierre Francois, and Emile Aben. On The Diversity of Interdomain Routing in Africa. In *International Conference on Passive and Active Network Measurement (PAM)*. Springer, 2015.

[171] Roger Dingledine. The lifecycle of a new relay. `https://blog.torproject.org/lifecycle-new-relay`, 2013. Last visit: 2019-12-23.

[172] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Symposium on Network and Distributed System Security (NDSS)*. The Internet Society, 2019.