University of
South-Eastern Norway

Faculty of Technology, Natural Sciences and Maritime Sciences

Master's Thesis in Computer Science

CS5000

Spring 2024

Saro Husaini

Circumventing Internet Censorship

University of South-Eastern Norway
Faculty of Technology, Natural Sciences and Maritime Sciences
Department of Science and Industry Systems
PO Box 235
NO-3603 Kongsberg, Norway

http://www.usn.no

This thesis is worth 60 study points

University of South-Eastern Norway

Circumventing Internet Censorship

Master's Thesis in Computer Science

Saro Husaini

Supervisor
Mohsen Toorani

University of South-Eastern Norway
Faculty of Technology, Natural Sciences and Maritime Sciences
Department of Science and Industry Systems
Campus Kongsberg

May 2024

Abstract

In the digital age, the struggle for information freedom faces significant challenges due to the sophisticated measures deployed by various governments to control and censor Internet access. A critical challenge emerges to the Internet's foundational principle as an open and free information exchange and communication platform. This global trend reflects diverse political, cultural, and social motivations behind Internet censorship and signifies a growing concern regarding the infringement of digital rights and freedoms. This thesis presents an in-depth investigation into Internet censorship techniques' dynamic and evolving landscape and the development of methods to circumvent them. Through a comprehensive literature review of Internet censorship techniques and circumvention technologies, we evaluated the effectiveness of existing tools and proposed our idea to enhance the circumvention efforts and contribute to the field of study.

We categorized an array of censorship techniques, highlighting their geographical variance and technical sophistication, underscoring the relentless arms race between censors and activists, in which technological advancements continually reshape the battleground. We answered research questions regarding the prevalent Internet censorship techniques used globally and available techniques for circumventing them. We addressed the effectiveness of the state-of-the-art Internet censorship circumvention methods against advanced censorship systems and how emerging technologies such as blockchain and AI can be leveraged to develop new circumvention methods.

We introduced ProxyPro, a system design that integrates multiple state-of-the-art techniques, offering a sensible solution to bypass Internet censorship effectively. Our approach combines the user's social graphs protection and enumeration resistance of the Lox proxy distribution system, the probe resistance and disguising abilities of our implementation of a probe-resistant proxy service, and NetShuffle's constant proxy shuffling strategy to achieve a system that is resistant to replay attacks, enumeration, and blocking while maintaining focus on simplicity, ease of implementation, and cost-effectiveness.

Acknowledgements

I express gratitude to my supervisor, Dr. Mohsen Toorani, for his invaluable advice and guidance. I deeply appreciate his continuous support and unwavering commitment.

I am grateful to my parents and my brothers for their full-hearted support and encouragement throughout this journey.

Contents

List of Figures

List of Tables

Acronyms

DNS	Domain Name System
SNI	Server Name Indication
VPN	Virtual Private Network
HTTPS	Hypertext Transfer Protocol Secure
TLS	Transport Layer Security
IP	Internet Protocol
AI	Artificial Intelligence
URL	Uniform Resource Locator
CDN	Content Distribution Network
CAs	Certificate Authorities
TCP	Transmission Control Protocol
ISP	Internet Service Provider
DPI	Deep Packet Inspection
GFW	The Great Firewall of China
QoS	Quality of Service
DDoS	Distributed Denial-of-Service
UDP	User Datagram Protocol
SSH	Secure Shell
DoT	DNS over TLS
DoH	DNS over HTTPS
BGP	Border Gateway Protocol
NAT	Network Address Translation
OONI	Open Observatory of Network Interference

1 Introduction

With the ever-growing Internet user base worldwide, authoritarian regimes and even less authoritarian governments are concerned about citizen movements and Internet activities. Consequently, they are enhancing their censorship tools and methods to limit or monitor user's access to certain websites and parts of the Internet. This censorship and surveillance can become a significant issue when some governments and organizations use it to infringe upon their citizen's basic human rights, such as feeling safe while using the Internet without intimidation.
Several research studies and solutions have been developed to circumvent Internet censorship, aiming to counteract the methods and tools used to restrict and censor the Internet. The struggle against Internet censorship started from the early days of wide adaptation of the Internet by the public, and there are studies dating back to the mid-1990s and early 2000s addressing Internet censorship and circumvention methods [1–3]. Despite significant efforts to counteract censorship mechanisms and regimes, the development of Internet censorship techniques consistently outpaces the progress made by activists and organizations dedicated to fighting censorship.
Numerous factors highlight the critical importance of the persistent effort against censorship from all parties involved. They are outlined as follows:

- **Technological Evolution and Adaptation:** As technology evolves, so do censorship methods and the tools to circumvent it. This constant tug of war necessitates continuous research and development of new circumvention technologies to stay ahead of restrictive measures. Adapting to changes swiftly and developing innovative solutions to overcome newer censorship forms is vital.
- **Geopolitical Variability:** Internet censorship varies significantly across different countries and regions and is influenced by political, cultural, and social factors. Understanding these diverse contexts is crucial for developing practical circumvention tools that are versatile and can be tailored to specific environments. Activists and researchers must consider these geopolitical nuances to create more universally applicable solutions.
- **Digital Rights and Freedom of Expression:** The fundamental right of access to information and freedom of expression is at the core of the struggle against Internet censorship. These rights' importance in fostering democracy, accountability, and societal progress underscores the necessity of combating censorship. This struggle is not solely about technology but also about upholding and promoting human rights in the digital age.

1.1 Research Questions

Internet censorship techniques and methods are continuously changing and rapidly advancing [4]. The censors are developing more active systems and introducing sophisticated methods to limit the effectiveness of any newly available circumvention systems. Researchers and developers are facing a difficult task in keeping up with the rapid deployment of censorship techniques worldwide. Catching up with the fast-growing censorship techniques, understanding the in-depth inner workings of several techniques, and coming up with effective circumvention proposals and solutions is challenging. The research into the chosen literature and existing work helps us understand the issues and identify research gaps in censorship circumvention, which led to the formulation of the following research questions:

1. What are the prevalent Internet censorship techniques used globally, and how effective are they?

   To answer this research question, we explored and investigated these globally widespread Internet censorship techniques in Chapter 2 and presented a table detailing the techniques alongside the countries where they are predominantly employed.

2. What are the widely available techniques for circumventing Internet censorship worldwide, and how effective are they?

   In the literature review in Chapter 3, we conducted an in-depth review and analysis of a wide range of cutting-edge circumvention techniques and their effectiveness. This thorough exploration provided us with insights and knowledge in response to this research question.

3. How can existing techniques for circumventing Internet censorship be combined to create a more powerful technique?

   The design we present in Chapter 5 represents our effort to offer a well-considered response to the research question posed.

1.2 Research Methodology

This section presents how we conducted our research, detailing the selected research method and design choices, including the selected research philosophy, research approach, and research strategy. We took the route of *qualitative* research approach for this thesis. We chose this design method to comprehensively understand previous work, ideas, and solutions from the collected sources and literature, explore the under-researched problems, generate ideas, and answer our research questions. Additionally, we use the data to understand and compare the effectiveness of

Internet censorship and circumvention techniques, analyze limitations, and evaluate advantages and disadvantages. As for the type of sources used in this thesis, mainly *secondary research* approach was utilized to study secondary sources, including academic papers, other dissertations, journals, and books that provide second-hand data and include descriptions, analysis, and research that have been sourced from primary data.

We needed flexibility in research to achieve an effective design, implementation, and evaluation. We followed an inherently problem-centered philosophy that values research that addresses real-world problems exactly as needed in our case. Our philosophy mainly aligns with *Pragmatism* research philosophy that prioritizes the research question over strict adherence to either qualitative or quantitative methodologies. This makes it highly suitable for interdisciplinary research where the nature of the problem dictates the research methodology. Pragmatism is especially relevant for applied research in computer science, such as software development and implementing information systems in specific social or organizational contexts. It supports the development of technology grounded in users' practical needs and experiences.

We are researching techniques and systems, presenting more up-to-date solutions, and improving existing techniques. A *systems development strategy* is the method that mainly aligns with our approach. It is widely employed in computer science and generally aims to improve existing systems or design, develop, and implement new ones. This research strategy served as our road map to investigate existing techniques and identify goals, requirements, and constraints. The reason for following this strategy is that system development focuses on creating practical solutions to real-world problems. It allows us to directly address users' needs or gaps in technology by developing systems that offer tangible benefits. It provides a structured and systematic approach to efficient development.

Given we collected most of our literature in a single period, the research mainly took the form of *cross-sectional time horizon* research. The research required us to collect a significant amount of papers and literature in a short amount of time through multiple online databases, which made the cross-sectional reasonable, fast, and reliable. Furthermore, the nature of our study pushed us to collect and read extra papers and expand our body of knowledge on the works and techniques we were researching even after the initial phase of the literature review. This additional effort also helps to mitigate the limitations typically associated with data collection confined to a single period inherent in cross-sectional research.

1.2.1 Literature Collection and Filtering

This subsection presents the frameworks and techniques for collecting, filtering, and analyzing papers and literature. We provided Table 1 of the keywords used, showing how the literature was targeted and the relevance of each keyword to our research topic. We selected the most effective keywords to optimize our search results and focused on the most recent publications. Table 2 outlines the sources of literature used, listing various publishers and the count of papers sourced from each, which helps demonstrate the diversity and scope of the literature review. Table 3 outlines the criteria for selecting and excluding papers for the literature review, such as peer-review status, citation count, and relevance to the research topic.

The tables demonstrate our commitment to quality by including primarily peer-reviewed scientific papers from reputable publishers. Furthermore, we prioritized research papers with a high number of citations, indicating trustworthiness and recognition within the research community. We also excluded papers that were not relevant to our core topic and those that were duplicative.

Table 1: The list of keywords used for the search along with the number of hits in Google Scholar from 2019 and upward.

Keywords	**Hits (Google) ≤ 2019**
Circumventing Internet Censorship	26
Internet Censorship Circumvention	83
Bypassing Internet Censorship	15
Bypass Censorship	505
Censorship Circumvention	645
Censorship Circumvention Tools	126
Internet Censorship	6,210

Table 2: The list of publishers and the number of their featured papers in this thesis.

Publisher	**Number of Paper**
The ACM Digital Library	21
The USENIX Association	16
IEEE Computer Society	10
The Internet Engineering Task Force (IETF)	6
The Internet Society	5
Springer	4
Privacy Enhancing Technologies Symposium (PETS)	4
Open Observatory of Network Interference (OONI)	2
Other publishing organizations and universities	14

Table 3: Features of chosen and excluded research papers.

Criterion	Details
Preference	• ~ 97% scientific research papers are peer-reviewed. • ~ 82% scientific research papers have more than 10 citations. • The Majority of chosen research papers are from 2017 and later.
Exclusion	• The scientific research papers which were irrelevant to the main subject. • The scientific research papers which were not written in English. • Repetitive Publications.

1.3 Contributions

The contributions and outcomes of this thesis are:

- A literature review on Internet censorship techniques and methods used worldwide.
- A literature review on the state-of-the-art Internet censorship circumvention tools and techniques.
- A proposed scheme and implementation contributing to Internet censorship circumvention efforts.

1.4 Outline

The rest of the thesis is structured as follows:

- **Chapter 2** presents a literature review of the current Internet censorship techniques employed by nation-state censors and specialized organizations to control and limit access to the free Internet.
- **Chapter 3** provides a literature review of the latest advancements and state-of-the-art Internet circumvention systems accompanied by a table that lays out each system's compact description, strengths, and weaknesses.
- **Chapter 4** details the components, design, and workflow of the various techniques integrated into the proposed scheme.
- **Chapter 5** introduces ProxyPro, our proposed design that combines several state-of-the-art circumvention techniques to provide an up-to-date and improved censorship circumvention technique.
- **Chapter 6** presents a security analysis and explores various attacks and vulnerabilities that could face our design and possible mitigation mechanisms.
- **Chapter 7** presents the implementation of our probe-resistant proxy, detailing the integrated security features and the outcomes of conducted tests.
- **Chapter 8** serves as the thesis's conclusion, summarizing the study's findings, particularly spotlighting ProxyPro, our proposed circumvention design. Additionally, it details future research suggestions and our future aims.

2 Internet Censorship Techniques

Internet censorship techniques can generally be classified into the three main elements of *prescription*, *identification*, and *interference* [5]. Prescription refers to the methodology censors utilize to decide which content categories to restrict, such as marking adult content sites as inappropriate. Identification involves the process whereby censors pinpoint specific traffic or identifiers for suppression or degradation, such as targeting webpages that incorporate the word "freedom" in an HTTP Header or those that receive traffic through the URL www.freedom.example, categorizing them as potentially objectionable. Interference describes the action taken by censors to intervene in communications, obstructing access to banned content by either completely blocking entry or diminishing the quality of the connection, for instance, deploying a technological solution that detects HTTP headers or URLs and ensures they are either entirely or partially blocked from access.
This chapter details Internet censorship techniques and explores their mechanisms. Furthermore, we provide a table detailing the techniques alongside the countries where they are predominantly employed.

2.1 TLS-based Filtering

Transport Layer Security protocol safeguards a communication's content by encrypting it, but it exposes the domain name of the requested resource. This domain name, defined during the TLS handshake process by Server Name Indication filed in an unencrypted client Hello message, still allows censors to observe and monitor the destination domain. Censors, by disrupting the TLS handshake or negotiation process, can effectively block communication attempts [6]. Censors are deploying various techniques for censoring TLS and, to a larger extent, HTTPS censoring [7]. The techniques include encrypted SNI, censored SNI, or omitted SNI, mostly associated with the SNI field. Censors are also able to censor HTTPS content via server certificates. In the case of TLS-based Filtering, Master et al. [8] observed in their survey of 70 nations a significant 41% of countries are employing TLS-based filtering techniques to restrict HTTPS traffic. This trend likely stems from the pervasive adoption of TLS encryption.

2.2 SNI-based Filtering

There are possibly several virtual servers hosted at a given network address by real servers in an encrypted connection using TLS. In the client hello message, the client must identify which domain name it wants to connect to using the SNI TLS extension [9]. In TCP-based TLS, the client hello message is unencrypted. QUIC is used to encrypt the client hello message; still, the confidentiality of the message is not effectively guaranteed and protected because the initial encryption keys are derived using a visible value on the wire [5]. Filtering software and censors can use SNI as a basis for filtering, impairment, or blocking since SNI is often sent in the clear. This is achieved by connections dropping of domains that match prohibited content [10]. There are continuous efforts for the standardization of SNI encryption and Encrypted-SNI as a new extension for TLS 1.3 has been deployed to most web browsers and more than 1 million Alexa top sites to prevent censors from learning the server names and the use of ESNI is showing promising results in some countries [11].

2.3 ESNI-based Censorship

The encryption of SNI is the main logical solution, as the SNI field is prone to data leakage, and the Encrypted Client Hello (ECH) is provided by TLS 1.3 [5]. The ESNI extension was put in place to protect and prevent data leakage caused by SNI before the appearance of ECH, which only encrypted the SNI field. Unfortunately, connections that specifically use ESNI extension for censorship can be targeted by censors. By this, the censor guarantees overblocking, but if ESNI is not yet widely deployed within the country, the overblocking can be worth the cost for the censor. Encrypted client hello is going to be an emerging standard for entire TLS client hello protecting when it is widely deployed. It fixes and prevents server name and hostname leakage similar to the ones of TLS 1.2 and non-encrypted DNS traffic [12]. Along with the encrypted DNS channel, it helps to unblock several websites currently censored in China, but unfortunately, China began censoring all uses of ESNI in 2020, even for unoffending connections [11]. The mechanism China uses to censor ESNI connections differs from the one used to censor SNI-based connections. This can suggest the deployment of new middleboxes specifically to target ESNI connections [5].

2.4 Omitted-SNI Censorship

Users have learned to exclude the SNI extension completely. This approach came to be called omitted-SNI. This approach restricts the amount of information that is available to a censor. Censors can block any connection that omits the SNI, just like ESNI. Even though this also risks overblocking. Censors blocking the connections that omitted the SNI field were observed by researchers [5].

2.5 Utilizing Certificate Response Fields

The server responds with the TLS certificate during the TLS handshake. The certificate contains the domain to which the user wants to access the TLS certificate [7]. This can create another easy course of action for censors to apply censorship. This method will not affect TLS 1.3, as all messages exchanged after the ServerHello in a TLS handshake are now encrypted. The EncryptedExtensions message, which was recently introduced, provides confidentiality protection to various extensions previously sent in the clear in the ServerHello [13].

2.6 Utilizing Content Distributors

Several governments pressure ISPs, content providers, social media, and websites to censor information or censor themselves, or they pass laws and legal frameworks within which ISPs and content distributors are urged or forced to follow the preferences of content restriction of the censor [14]. The extent of this censorship can reach from the services that provide online storage to the installed software locally. Content distributors commonly use keyword identification on their platforms to detect restricted terms. Censors and governments can provide terms on those types of keyword lists to content distributors, or it is even expected that content distributors compile their keyword lists based on the preferences of the censor [5]. As for censoring videos and images, censors commonly require content providers to use hash matching to detect and censor unwanted visual and audio media [15].

2.7 DPI-based Censorship

Deep packet inspection allows the real-time monitoring of the content of data flows by Internet service providers and gives them the ability to make decisions about how to handle them accordingly; this became possible with the technological advances and the introduction of the intelligence DPI to routers and network monitoring equipment [16]. DPI is often used for keyword identification because, unlike other techniques that only examine the application headers, it reassembles network flows to allow the application "data" subsection to be examined. DPI can also leverage flow characteristics and additional packets, differentiating it from other identification technologies (e.g., timing and packet sizes) while identifying content. While it continues routing the original packets, DPI normally analyzes a copy of data to prevent any substantial impact on the quality of service. An Intrusion Detection System configured for censorship running on a machine cluster commonly analyzes the split traffic using a fiber splitter or mirror switch [5].

2.8 HTTP Request and Response Header Identification

The client program and server send and receive a list of strings on every HTTP request and response called HTTP headers. An HTTP header includes a lot of helpful data for traffic identification [5]. The HTTP method field is needed to accomplish anything useful, even though in HTTP/1.1 and later, the *host* is in the HTTP request header is the only required field. As such, the two fields most often used for censorship are *method* and *host.* A censor can sniff targeted traffic and commonly identify the page name (GET /page) and a particular domain name (host). Censors can use a combination of Transport Header Identification and HTTP Request Identification to filter specific URLs.

Information sent in response by the server to the client is used in the response identification to identify undesirable content, in contrast to the HTTP Request Header Identification, which is reliant on the data in the HTTP request from the client to the server. Park et al. [17] demonstrated in 2009 that the GFW had used this technique. However, the GFW discontinued this practice during Park's study. Keyword filtering over TCP streams is the technique that most censors probably rely on instead of HTTP response filtering due to the overlap between keyword filtering and HTTP response filtering.

Unlike HTTP response filtering, *Keyword Filtering* is a common technique that is widely used in Internet censorship to restrict access to unwanted online content based on the presence of specific keywords or phrases [5]. The GFW is a prominent example of understanding the mechanism of keyword filtering. It is one of the most active and sophisticated systems in Internet censorship; new keywords are constantly added, and the list is updated primarily based on recent events and controversies. The GFW achieves keyword filtering by searching the unencrypted packet streams for forbidden keywords. When undesirable keywords are detected, it disrupts an offending stream by TCP RST packet injection, and further communication between the involved hosts is blocked for a few minutes. Weinberg et al. [18] investigated the GFW's application layer, and their understanding of HTTP reveals that forbidden keyword detection only applies to specific locations within an HTTP request. Requests containing the English word "search" are subject to more extensive inspection for forbidden keywords than those without. Additionally, observations of changes in the forbidden keyword list since 2014 revealed a significant rate of change, with over 85% of the keywords being replaced. The remaining keywords address unchanged sensitive topics, while the newly added keywords mainly relate to recent events and controversies. A "penalty box" period is another important study finding. During a 90-second window following the disruption of a TCP stream, all subsequent requests initiated from the same client to the same server face a 50–75% probability of being blocked, even if they do not contain any censored keywords. This action increases the overall effectiveness of this type of keyword filtering censorship.

2.9 IP Blocking

Internet service providers under their governments' control can block the IP addresses of targeted websites so that regular users cannot access them [19]. The user's request for an IP address is monitored and compared against the censor's list of blacklisted IP addresses. The ISP drops the connections, trying to reach unwanted and forbidden websites. IP blocking can occur using IP addresses contained in IPv6/IPv4 headers, and a method that censors can use to identify IP addresses is Transport Header Identification, taking advantage of a few transparent pieces of information contained in the Transport headers, including the destination and source IP address. This allows the censor to block unwanted content via IP blacklisting. [5].

2.10 Protocol Identification

Censors use different techniques for protocol identification. A basic method involves identifying traffic by the standard ports used by protocols (e.g., recognizing all TCP traffic as HTTPS). However, more sophisticated methods analyze the properties of payload data and the traffic flow behavior, which can identify protocols even when they don't use their standard ports [20]. Some countries, like Iran, deliberately impair the performance of secure protocols such as HTTPS. HTTPS encrypts data to prevent interception and analysis. By degrading its performance, users are encouraged to switch to less secure protocols like HTTP, which can be easily monitored and analyzed by censors [21].

Countries with stringent censorship regimes, such as China, invest in identifying protocols used by censorship circumvention tools. Detecting these tools allows censors to block them effectively, which has led to an arms race between censors and tool developers. This is an advanced technique where the censor attempts to initiate communication with a host using the suspected circumvention protocol. If the connection is successfully established, it confirms the host's circumvention tool use. China has notably used active scanning to block access to Tor, a network that protects users' privacy and anonymity by routing their Internet traffic through a series of relays [22].

2.11 DNS Tampering

Censors have various mechanisms to block or filter access to content by altering DNS responses. These methods include blocking the response, replying with an error message, or manipulating DNS records to return an incorrect IP address, preventing users from visiting the requested website. Encrypted transports for DNS queries, such as DNS-over-HTTPS and DNS-over-TLS, can mitigate interference with DNS queries between the stub and the resolver [23, 24]. Responding to a DNS query with an incorrect address can be achieved through on-path interception, off-path cache poisoning, and deception by the nameserver.

In countries where regimes exercise control over domain name servers by *deregistering* the domain, they can effectively render a website hosting objectionable content invisible to Internet users. This action disrupts the translation of domain names into their corresponding IP addresses, making the website inaccessible to users' browsers and effectively silencing the targeted content.

Jin L et al. [25] performed 7.4 million DNS lookup measurements on 75 DoH resolvers and 3,813 DoT and identified that 1.42% of DoH and 1.66% of DoT responses responses are subject to DNS manipulation. Furthermore, their findings revealed that over two-thirds of DoT and DoH resolvers manipulate DNS responses for at least one domain, suggesting that DNS manipulation is pervasive in encrypted DNS, potentially facilitating the advancement of Internet censorship and increasing the effectiveness of DNS Tampering.

2.12 Performance Impairment

Unlike traditional censorship tactics that completely prevent users from accessing certain websites, services, or content, the strategy here involves deliberately reducing the network connection quality to these destinations [26]. This doesn't stop access; instead, it makes the experience of using these sites or services frustratingly slow or unreliable. This kind of censorship aims to sour the user experience to the point where individuals choose not to use a specific site or service. They might switch to a different, possibly less secure or less private, platform that doesn't face these restrictions, or they might forego using the Internet for these purposes altogether if no better alternatives exist. This indirect approach can be very effective because it doesn't draw as much attention or backlash as blocking access. Yet, it discourages or limits the use of certain online services or the dissemination of particular types of information. Iran has throttled the bandwidth for HTTPS traffic. By slowing down HTTPS, the authorities might be pushing users towards using HTTP instead, which is unencrypted and allows for easier surveillance and control of Internet traffic [21].

2.13 Packet Dropping

Packet dropping is a straightforward technique used in the digital censorship toolkit to disrupt or completely halt the flow of information deemed undesirable by a censor. The essence of packet dropping is precisely what the term suggests: the selective identification and discarding of data packets as they travel across the network. When a censor, which could be a government or any controlling authority, identifies traffic it considers unwanted, it simply refuses to forward the associated data packets according to the standard procedures outlined by network protocols. Packet dropping can be effectively combined with other censorship mechanisms, provided the censor controls a critical point in the network infrastructure, such as a key router, through which the user's traffic must pass. This control point allows the censor to monitor and manipulate traffic flow as needed [5].

2.14 RST Packet Injection

RST Packet Injection is a specialized form of packet injection that focuses on exploiting the TCP communication protocol [17]. TCP is a foundational protocol in the Internet protocol suite, ensuring ordered, reliable, and error-checked delivery of data between applications running on hosts across the IP network. An RST packet is a normal part of TCP's control mechanism that abruptly closes a TCP connection. The packet indicates that one side of a conversation should stop sending data, prompting the recipient to terminate the connection. In an RST Packet Injection attack, an attacker sends RST packets to both ends of a TCP connection, impersonating each side of the conversation to the other. The communication session is closed because each end believes the other has terminated the connection. This technique can disrupt any ongoing data exchange, ranging from web browsing sessions to file transfers, making it a powerful tool for censorship or sabotage.

2.15 Full Internet Disconnection

One of the most extreme forms of Internet censorship is the complete shutdown of network access within a specific region or country. Unlike other censorship techniques that target specific websites, services, or types of traffic, this method involves cutting off all Internet access. This is typically achieved by manipulating the Border Gateway Protocol, which is a crucial mechanism that ISPs use to route Internet traffic across the world [5]. By withdrawing BGP prefixes, the identifiers that allow routers to find the best paths to network destinations within the censoring entity's jurisdiction, the authority effectively makes it impossible for data to enter or leave the controlled region through normal Internet pathways.

2.16 BGP Hijacking

BGP hijacking is a sophisticated method of Internet censorship and manipulation that involves the unauthorized takeover of groups of IP addresses by misdirecting Border Gateway Protocol

routes [27]. BGP is a protocol used to make decisions about routing Internet traffic, relying on announcements (or "advertisements") of available IP routes. These routes guide data packets across the Internet, determining how information travels from its source to its destination.
In BGP hijacking, an entity, typically a malicious actor or a government agency, makes false BGP announcements. These incorrect announcements can cause Internet traffic to be rerouted through unintended pathways, allowing the hijacker to intercept, censor, or disrupt the flow of information [5]. This can be done in two main ways:

1. **Within a Jurisdiction:** The hijacker announces incorrect routes that are meant to stay within a certain region or country. This can prevent users within the jurisdiction from accessing information outside of it, as their Internet traffic is misdirected to nowhere or to a controlled endpoint within the jurisdiction.

2. **Beyond a Jurisdiction:** The hijacker announces bogus routes to the global Internet, affecting traffic worldwide. This can lead to international users being unable to access content or services hosted in the hijacked IP ranges, or it can divert global traffic through the hijacker's networks, allowing for surveillance or data interception on a massive scale.

2.17 Distributed Denial of Service (DDoS)

Distributed Denial of Service attacks are a form of attack in which multiple compromised systems or computers are used to target a single system infected with a Trojan, causing a Denial of Service [28]. The aim of a DDoS attack is to overwhelm the targeted system's resources, making it unable to respond to legitimate traffic or to crash it altogether. These attacks can be motivated by various reasons, ranging from political to purely malicious intent. There are mainly two categories of impacts from DDoS attacks [5]:

1. **Flood Attack:** This involves overwhelming the target with excessive traffic to the point where the service becomes unusable. The service is bombarded with so many requests that it can't handle legitimate user traffic, leading to a denial of service for intended users. The resources of the targeted service are consumed by dealing with the flood, rendering the service ineffective.

2. **Crash Attack:** The goal here is to exploit vulnerabilities in the target system to cause it to crash. Once the service crashes, its resources can be reallocated or left unused. Unlike the flood attack, where the service is simply overwhelmed, a crash attack seeks to exploit specific weaknesses that can bring the service down completely.

2.18 Multi-layered Approach

A multi-layered approach controls information access on the Internet by simultaneously employing various censorship techniques. This strategy ensures that if one method fails or is circumvented, others remain in place to maintain control over the flow of information [5]. It's akin to the principle of in-depth defense in cybersecurity, where multiple layers of security measures are deployed to protect against threats. There are several ways in which censorship in depth can be implemented:

1. **Blocking Content Through Multiple Techniques:** This involves using different methods to block the same content. For example, a censor might block access to a website by interfering with its DNS entry, blocking its IP address, and filtering out HTTP requests to the site. This makes it much harder for users to bypass censorship since they would need to overcome multiple barriers.

2. **Deploying Parallel Systems:** To improve the reliability of censorship efforts, authorities might deploy several different systems designed to block access to the same domains or content. This redundancy ensures that even if one system fails or is bypassed, others are in place to continue the censorship.

3. **Using Complementary Systems to Limit Evasion:** Censors may block or severely limit the use of certain protocols, pushing users towards using other protocols that are easier for the censorship mechanisms to filter. For example, blocking encrypted protocols could force users to rely on unencrypted protocols, which can be monitored and filtered more easily.

2.19 Manual Filtering

Manual filtering is a censorship method that relies on human effort rather than automated systems to identify and control content deemed inappropriate or undesirable by certain standards, often used by governments or organizations to suppress dissent or regulate online discourse [5]. Unlike automated techniques that use algorithms to build blocklists targeting specific IPs or DNS records, manual filtering involves individuals directly reviewing, removing, or flagging content such as websites, blogs, articles, and other forms of media. This method allows for a more nuanced understanding of the content, potentially reducing the errors common in automatic filtering systems, such as misidentification or overclocking. Manual filtering can be implemented at various levels of the Internet infrastructure. It can occur at the backbone/ISP level, where large volumes of traffic are monitored and filtered, or at an institutional level, affecting specific organizations or content providers [5].

Table 4: Censorship techniques used in different countries.

Censorship technique	Countries
DPI-based Censorship	• The Great Firewall of China was explored by several researchers, and they found evidence of the use of DPI to censor content and tools [17, 29, 30]. • DPI was used by China, Iran, Ethiopia, and other countries to block the obfs2 protocol. • Malaysia has been accused of using targeted DPI and DDoS attacks to identify and subsequently attack materials of pro-opposition parties [5].
Keyword Identification	• 516 keyword combinations were censored on WeChat group chat in China, which were directly related to COVID-19 between January 1 and February 15, 2020 [31].
Utilizing Certificate Response Fields	• The use of certificate response fields to censor connections was observed by the Reliance Jio ISP in India [12].
Omitted-SNI Censorship	• In Russia, censors blocking connections that omit the SNI field were observed [5].
ESNI-based Censorship	• China began censoring all uses of ESNI in 2020, even for unoffending connections [5].
SNI-based Filtering	• SNI-based filtering products are provided by several security companies [10], and the widespread SNI blocking and filtering was detected in many countries, including Asian countries like China, Iran, Qatar, UAE, South Korea, Turkmenistan, and others like Egypt and Turkey [5, 11, 32]. • In Russia in March 2022, SNI blocking against QUIC traffic was first detected [5].

TLS-based Filtering	• It was observed that in a survey of 70 nations, 41% of countries employ TLS-based filtering techniques to restrict HTTPS traffic [8]. • China, Ethiopia, and Iran are noticeable countries where TLS-based filtering was or is employed [33]. • The use of TLS filtering and SNI-based blocking was also observed in India [34].
HTTP Request Header Identification	• Bahrain, Bangladesh, China, Iran, India, Malaysia, Pakistan, Saudi Arabia, Thailand, South Korea, Russia, and Turkey [21,35,36].
IP Blocking	• Master et al. [8] showed in their study the current and historical use of IP blocking in most of the countries that are notorious for Internet censorship and not considered free in terms of Internet freedom [37], including China, Iran, Myanmar, Saudi Arabia, Egypt, Ethiopia, Pakistan, Russia, Belarus, Kazakhstan, Turkey, Azerbaijan.
Protocol Identification	• Iran used protocol identification to detect and throttle SSH traffic and implemented an allowlist protocol filter, permitting only DNS, TLS, and HTTP on specific ports in 2020 [5,38]. In 2020, they implemented an allowlist protocol filter, permitting only DNS, TLS, and HTTP on specific ports [38]. This approach censors unidentifiable connections, significantly restricting Internet access to approved protocols. • Comcast used protocol identification for traffic management by injecting TCP Reset (RST) packets into BitTorrent traffic streams. This disrupts BitTorrent traffic, a peer-to-peer file-sharing protocol, to manage network bandwidth and reduce congestion. • China employed protocol identification, specifically active scanning, to identify and block Tor relays [22]. • Russia in 2022 appeared to use protocol identification to block most HTTP/3 connections [39]. Blocking HTTP/3 could be an attempt to limit the use of more secure and efficient Internet protocols.

DNS Tampering	• Italy, France, Estonia, and Iceland employed DNS tampering to restrict access to content deemed illegal within their borders, and Spain manipulated DNS for blocking 25 websites during the 2017 Catalan referendum of independents [40]. • A survey of 43,000 DNS resolvers across 173 countries discovered that 26% of these resolvers in 109 countries were affected by DNS pollution, i.e., receiving spoofed IP addresses instead of genuine ones. Nearly 80 percent of the DNS resolvers tested were affected in South Korea [41]. • A paper on the measurement of global DNS Manipulation listed the top 10 countries by median percent of manipulated responses per resolver; Iran led the list with 6.02%, China second at 5.22%, followed by Indonesia, Greece, Mongolia, Iraq, Bermuda, Kazakhstan, and Belarus [42].
RST Packet Injection	• In 2007, Comcast employed RST Packet Injection to disrupt traffic deemed BitTorrent in the United States [5]. • China has utilized RST Packet Injection as a method of censorship, which notably impacted the functionality of encrypted or obfuscated protocols, including Tor's [22].
Packet Dropping	• The Great Firewall of China employs packet dropping as a fundamental method of technical censorship [5]. • Iran has applied packet dropping as a method to throttle SSH traffic, which is often used for secure remote server access and encrypted file transfers [21]. • In Iran, India, Russia, and Uganda, packet dropping was observed during the handshake or working connection for QUIC traffic [19].
Performance Impairment	• Iran has been reported to manage the bandwidth allocated to HTTPS traffic, forcing the use of unencrypted HTTP traffic [5].

Full Internet Disconnection	• In 2007, Myanmar used network disconnection to suppress a rebellion. • In 2009, China severed network connections in the Xinjiang region during periods of unrest, aiming to prevent the spread of protests to other areas. • The Arab Spring witnessed the most extensive application of network disconnections, with significant events in Egypt and Libya in 2011 and Syria in 2012. • In April 2019, Russia temporarily disconnected Russian networks from the global Internet to test the country's network resilience, and this required Russian telecom companies to divert all Internet traffic through state-controlled monitoring points. • India experienced the highest frequency of Internet shutdowns in 2016 and 2017, leading the world in such measures [5].
BGP Hijacking	• In 2008, the Pakistani government censored YouTube by altering the site's Border Gateway Protocol (BGP) routes. • In 2018, the state-owned ISP China Telecom hijacked millions of Google's IP addresses. This was similar to the BGP hijacking of websites belonging to the US government by the same ISP in 2010. • ISPs in Russia in 2022 and Myanmar in 2021 have made multiple attempts to hijack the BGP prefix associated with Twitter [5].
Distributed Denial of Service (DDoS)	• In 2012, the UK's Government Communications Headquarters deployed DDoS attacks to temporarily disable Internet Relay Chat (IRC) channels used by members of Anonymous. • Websites expressing dissent often face DDoS attacks during politically charged moments, as evidenced by incidents in Burma. • Authorities in Russia, Zimbabwe, and Malaysia have been implicated in disrupting oppositions through DDoS attacks around election times. • In 2015, China initiated DDoS attacks employing a true Man-In-The-Middle framework integrated with the GFW [5].

Notice and Takedown	• Google used mechanisms to adhere to the European Union’s "Right to be Forgotten" regulations, the liability stipulations for electronic platform providers. • In the USA, copyright-focused notice and takedown procedures are established under subsection 512 of the United States Digital Millennium Copyright Act (DMCA) [5].
Manual Filtering	• China’s extensive monitoring workforce is a prime example of manual filtering. Internet Content Providers, like Google or Weibo, must obtain a business license to operate within China, making ICPs accountable to the Chinese government for inappropriate content [5].

3 Internet Censorship Circumvention Techniques

This chapter provides an in-depth review and analysis of a wide range of cutting-edge circumvention techniques and their effectiveness. Furthermore, we provide a table detailing the techniques' strengths and weaknesses. The classification of circumvention techniques in this chapter is primarily based on their most distinguishing feature, leading to the possibility of categorizing some systems in different sections despite sharing similar methods and protocols with each other [i].
Several research studies and solutions have been developed to circumvent Internet censorship, aiming to counteract the methods and tools used to restrict and censor the Internet. Some methods and strategies prove more effective than others, and certain approaches may become outdated over time. Early solutions like *Infranet* [43] enabled users to circumvent Internet censorship by establishing covert channels with accessible web servers. Requesters compose secret messages using hard-to-detect sequences of requests, and Infranet responders covertly embed data in the openly returned content. Servers provide clients access to censored sites while continuing to host normal uncensored content. However, Infranet had some downsides; Infranet requester software was to be distributed on physical copies such as CD-ROMs. Moreover, this type of distribution mechanism is slow, provides evidence for culpability, and is easy for censoring governments to control. The Infranet architecture could also not protect against an impersonation attack where the censor establishes an Infranet responder and discovers requesters by identifying the web clients that send meaningful Infranet requests.

3.1 Proxy-based Systems

Subsequent concepts introduced the idea of generating millions of short-lived proxies, where each proxy would only remain operational for a few minutes [44]. This strategy was realized through the assistance of volunteer websites outside the censored regions, eager to support unrestricted Internet access. These volunteer websites executed a JavaScript program that allowed them to function as temporary proxies by channeling traffic to and from the censored areas via the visitor's web browser. Once the visitor leaves the site, the proxy vanishes without leaving any remnants on the user's device. Nevertheless, this approach had its vulnerabilities; adversaries could potentially intercept the traffic, exposing the data being transmitted. Additionally, there was a risk of attackers accessing sensitive user information, such as IP addresses and port numbers, making it relatively straightforward to disrupt the proxy-based data transmission.
Another tool that utilizes servers located in countries with Internet freedom is *Camoufler* [45]. This

[i]For instance, automated censorship evasion tools are placed in the same category because automation is their defining characteristic, even though they might share the same tunneling method with VPNs.

tool leverages the Instant Messaging (IM) platform to route traffic that has been censored. Camoufler aims to ensure minimal delay, sufficient data transfer rates, dependability, and resistance to blocking by incorporating IM services into its core tunneling protocol. Users employ the Camoufler client to direct their requests to a Camoufler server in a jurisdiction without censorship; this server then retrieves and delivers the restricted content back to the user, effectively acting as a proxy. However, Camoufler's reliance on IM platforms introduces specific vulnerabilities [45]. If an adversary, particularly one who controls the IM platform used, seeks to block access to Camoufler's services, they could directly censor the IM IDs associated with Camoufler's servers [43]. Furthermore, there's an inherent risk of detection through user behavior analysis over time. Anomalies in messaging patterns, such as sending messages at unusual hours or deviations from typical behavior, could potentially alert adversaries to the use of Camoufler or similar tunneling services.

Utilizing an overlay network is an approach embodied by the *Social Network Friendship Enhanced Decentralized System (SEnD)* [46]. *SEnD* incorporates IP-over-P2P (IPOP) tunnels and operates fully decentralized. It empowers users in areas without censorship to serve as proxy servers for their social network friends in regions where censorship is prevalent, thereby facilitating uncensored access. This mechanism is enabled by establishing peer-to-peer virtual private IP tunnels connecting users directly to the proxy servers within their social circles. *SEnD* is designed to counteract censorship strategies like active probing attacks and IP address blocking, offering enhanced resilience over other existing solutions. It securely transmits traffic through a peer-to-peer model, encrypting messages within encapsulated formats, thus avoiding centralized control points. Moreover, SEnD capitalizes on trusted social connections to discover and forge pathways between those offering support and those in need. Utilizing dynamically allocated NATed endpoints for traffic proxies further obstructs efforts to systematically identify or probe these connections [46].

Current tools designed to bypass Internet censorship often encounter various challenges, such as susceptibility to censorship due to reliance on a limited number of IP addresses that can be easily identified and blocked, for instance, with VPNs, Tor, Psiphon, and Lantern [7]. These systems are also known for their high operational costs and suboptimal service quality. In contrast, *MassBrowser* [47] proposes an efficient solution for a broad user base to circumvent censorship, ensuring quality of service (QoS) at a lower operational cost. Operated by volunteers, *MassBrowser* is a proxy-based platform with user-friendly graphical interface software compatible across major operating systems. It employs a variety of evasion techniques, such as Domain fronting and CDN browsing, to establish its client-to-client proxy architecture. Unlike standard proxy-based systems that utilize public IP addresses, MassBrowser's proxies are located behind public NATs. This setup complicates attempts at blocking MassBrowser, as blocking its IP addresses could also inadvertently block other clients sharing those NATed IPs [45]. MassBrowser reduces operational costs by using shared IP addresses, effectively preventing IP enumeration and enabling handling a large volume of traffic through the cooperation of censored clients and volunteer proxies. It also allows users to adjust their privacy level, allowing for partial or full routing of their connections through MassBrowser's Tor interface for enhanced privacy. However, it's important to note that MassBrowser was not primarily designed with user anonymity in mind. Consequently, volunteers may have visibility into the web destinations their connected clients are accessing. Like other circumvention tools, MassBrowser's security could be compromised if censors intercept TLS communications [47].

3.2 TLS-based Tools

The Transport Layer Security protocol has emerged as a dominant protocol on the Internet, accounting for over 70% of web page loads in browsers like Mozilla Firefox [7]. Its widespread adoption has also made it a foundational component in various anti-censorship technologies, including Tor and Signal [7]. TLS effectively masks legitimate web traffic, offering a layer of protection against eavesdroppers by encrypting content. However, TLS alone is insufficient for completely evading censorship efforts. This is primarily because the initial stages of the TLS handshake are unencrypted, allowing censors to detect a client's support for specific encryption methods, key exchange algorithms, and extensions through the transparent "Client Hello" message. These vulnerabilities and new challenges have led to the repeated blocking of services like Tor and tools like Meek by firewall technologies such as Cyberoam and FortiGuard [48].

In response to these limitations, Frolov et al. [7] developed *uTLS*, a specialized TLS client library designed to offer granular control over TLS handshakes. uTLS grants developers the flexibility to choose any combination of cipher suites and extensions, enabling them to precisely replicate the behavior of other widely used TLS implementations. Furthermore, a dataset was merged with uTLS, providing developers with the capability to use code automatically generated on uTLS website. This facilitates the configuration of uTLS to accurately simulate the popular fingerprints that uTLS developers have documented. Despite these enhancements, uTLS is not without flaws. For instance, it might not fully accommodate client hello messages generated through automated fingerprinting processes. Additionally, the integrity of connections could be compromised if servers opt for a cipher suite not yet supported by uTLS, potentially leading to visible disruptions in the connection [7].

VPN services and Tor are notably effective in combating Internet censorship, as they conceal the user's intended destination domain from Internet Service Providers [7]. Nonetheless, a notable limitation of these approaches is that all the client's traffic is routed through the helper nodes, burdening these nodes with the task of allocating a substantial part of their network bandwidth to the client's traffic. To address this issue, *BlindTLS* [12] introduces a method for circumventing Internet censorship that achieves minimal additional burden by selectively routing packets through an encrypted TCP proxy while also concealing the true SNI within TLS 1.2. This strategy enhances user performance by maintaining data transfer efficiency and effectively navigating ISP-imposed content restrictions.

Given that HTTPS also encounters certain challenges due to its dependence on TLS, *HTTP/3* [19] was introduced in 2021 as the latest iteration of the HTTP protocol. It employs QUIC for its underlying encrypted transport, offering advantages such as reduced time for connection setup, continuous encryption, and decreased susceptibility to modifications or tampering by intermediary devices within the network [49]. To evaluate and compare the effectiveness of HTTPS and HTTP/3 in real-world applications, *OONI* deployed a censorship measurement tool equipped with an HTTP/3 module. This tool's findings showed that HTTP/3 requests often encounter fewer blocks than traditional HTTPS requests, with some instances of HTTP/3 requests not being blocked at all. The relatively low traffic volume attributed to QUIC and its novelty as a protocol may account for these observations. Nonetheless, HTTP/3 is not without its vulnerabilities. For instance, if an HTTPS request times out during a TCP handshake (TCP-hs-to), the HTTP/3 request also fails before the QUIC handshake completion. Furthermore, Elmenhorst et al. [19] identified UDP endpoint blocking as a tactic against HTTP/3 connections. Blocklisting IP addresses also poses a challenge for HTTP/3, as IP-based restrictions can hinder access to hosts that are on the blocklist.

Moreover, the utilization of SNI filtering as a censorship tool has been on the rise. ESNI extension for TLS 1.3 has been implemented across most web browsers and more than 1 million Alexa top sites to prevent censors from learning the server names and as a response to SNI filtering [7, 11]. The operational principle of ESNI is grounded in drafts that are evolving. Based on its third Internet draft, the ESNI mechanism typically involves the client acquiring a publicly accessible ESNIKey for the intended server through a trusted channel, often via a DNS TXT query through an encrypted DNS channel. Once the ESNIKey is acquired, the client generates an encryption key from it and a client-selected key [11]. This new key is then used to encrypt the server name in the TLS ClientHello message. Upon receiving this, the server decrypts the server name, and the ensuing connection proceeds as per the usual TLS 1.3 protocol. ESNI is designed to rectify and avert the leakage of server names and hostnames that are common with TLS 1.2 and unencrypted DNS queries. It also facilitates access to many websites currently blocked in regions like China through the use of encrypted DNS channels. Nevertheless, ESNI faces challenges, including vulnerability to blocking, evidenced by reports of GFW targeting all ESNI traffic since July 2020 and limited adoption. Hoang et al. [50] revealed that only between 1.5% and 2.25% of domains in Top-Level

Domain zone files possess a valid ESNI key, indicating a low uptake rate of ESNI.
The transmission of regular DNS traffic in unencrypted form exposes it to several vulnerabilities, such as the risk of privacy breaches [51]. Given the availability of protocols that encrypt DNS queries, evaluating their influence on Internet censorship is crucial. Research into the use of encrypted DNS resolvers to bypass Internet censorship has shown mixed results. For instance, it was found that 37% of domains blocked by censorship were accessible from certain locations within a specific country using encrypted DNS resolvers. However, in another country with similarly stringent censorship practices, encrypted DNS resolvers did not facilitate access to any blocked domains [25]. This suggests that the success of circumventing censorship with encrypted DNS resolvers can significantly differ depending on the country. Additionally, the choice of encrypted DNS resolver type at a given location does not substantially alter the accessibility of blocked domains. Despite this, adopting an encrypted version of the DNS protocol is crucial to mitigate potential spoofing attacks. DNS Queries over HTTPS (DoH) and DNS over TLS (DoT) are two widely recognized forms of encrypted DNS [52]. Both aim to secure DNS traffic by utilizing TLS, thereby providing an essential safeguard against interception and manipulation of DNS queries.

3.3 Blockchain-based Systems

MultiProxy [53] introduced a method for bypassing censorship through a blockchain-based economic model, establishing a Peer-to-Peer (P2P) system. This model ensures a harmonious exchange of resources and safeguards the anonymity of those initiating requests via multi-hop messaging. It helps with overcoming the shortcomings of the client-server-based architecture by combining the advantages of both client-server, resulting in latency nearly equivalent to that of a direct server connection, while also enhancing user privacy and security. Nevertheless, the system requires the implementation of various traffic obfuscation techniques to circumvent mechanisms like the GFW and would benefit from a more accessible, customizable graphical user interface (GUI) to replace its current command-line interface.
Another blockchain-based system that utilizes Bytecoin for censorship circumvention is Skywhisper [54]. It exploits the inherent redundancies in blockchain transactions to embed and transmit data securely and unnoticed. Skywhisper's system modifies cryptographic components within blockchain transactions, exploiting random elements in cryptographic signatures to embed hidden messages. This method ensures that the transactions appear normal to outside observers, thus maintaining their covert nature.

3.4 Automated Censorship Evasion Tools

Given the challenges of manually circumventing censorship, especially with the rapid evolution of censorship techniques, there is a significant need for automated censorship evasion tools like *Geneva (Genetic Evasion)* [55]. Geneva evolves packet manipulation-based censorship evasion strategies using a novel genetic algorithm against nation-state-level censors [55]. Geneva is also used to provide server-side censorship circumvention strategies [56]. Geneva was converted from client-side to server-side by configuring it to be able to train over a variety of applications across a variety of protocols, specifically giving support for HTTPS, FTP, DNS-over-TCP, and SMTP. This adaptation also involves configuring it to understand the meanings behind any packet header fields. Moreover, Geneva provides the ability to automatically discover censorship evasion strategies against on-path network censors. The genetic building blocks of Geneva allow it to both re-derive all previously published supported schemes alongside deriving altogether new strategies that prior work was not posited as effective. However, Geneva can fail to discover strategies when its access to header fields is heavily restricted [56]. Additionally, it is a relatively new technology that needs time for wider deployment.

Many Internet circumvention protocols, such as obfs4 [57], which rely on underlying TCP connections, consistently run the risk of experiencing connection throttling and dropping attacks targeting the underlying TCP connection. The *Turbo Tunnel* [58] censorship circumvention system addresses this issue by introducing a separate virtual session that outlives any TCP connection. This is achieved by reconnecting the client to another session if a TCP connection is throttled or terminated. The Turbo Tunnel design uses a session/reliability layer in the middle of the protocol stack between the user's application streams and the obfuscated network connection responsible for evasion. The session/reliability layer is not exposed to the censor; it is an internal layer. When the session/reliability layer transforms the outgoing streams into packets and vice-versa, packets are encapsulated and transmitted inside a blocking-resistant obfuscated tunnel. Moreover, the idea of leveraging the system to switch between different obfuscation strategies without losing the end-to-end session state provides more freedom and flexibility [58]. Nevertheless, Turbo Tunnel is an idea in its early stages; it is being presented as a general design pattern rather than a usable code or a reusable library. Furthermore, due to the lack of built-in cryptography, finding a suitable cryptographic protocol that works with the Turbo Tunnel model is needed.

3.5 VPNs

An Internet circumvention technique that is sometimes overlooked is the use of Virtual Private Networks. While ISPs can block VPNs once detected, VPNs still offer crucial security features such as confidentiality, integrity, and authentication for the traffic they encapsulate. Among the leading VPN technologies are WireGuard [59] and OpenVPN [6, 7]. WireGuard uses state-of-the-art ChaCha20-Poly1305 authenticated encryption and encapsulates IP packets into UDP [59]. WireGuard does not rely on standard TLS and uses its handshake implementation. The client and server also authenticate others with their public keys, an authentication model similar to SSH. On the other hand, OpenVPN provides IP or Ethernet encapsulation into TCP or UDP protocol. Unlike WireGuard, OpenVPN utilizes a TLS-like method for encryption, integrity checks, and authentication [6]. Its ability to hide TLS negotiation from eavesdropping by using a pre-shared key and providing full support for TLS handshake renders OpenVPN particularly resistant to censorship techniques.

3.6 Video Steganography Systems

An extra specialized yet important circumvention technique involves leveraging peer-to-peer and high-performing covert channels over WebRTC media streams [60]. WebRTC, as a project enabling text and video communications and real-time voice capabilities between web browsers and devices, is the target of censors to restrict the flow of data and inspect video content. To address this problem, *Stegozoa* [61] and similar tools were developed, aiming to prevent the ability of adversaries to detect covert data transmissions while they are in control of WebRTC gateways. Stegozoa embeds the covert data into the WebRTC video signal steganographically. It uses steganography techniques and applies them within the video coding pipeline of WebRTC, fine-tuning the implementation for efficient use of the available covert channel capacity and ensuring undetectability. By this, it aims to tackle the vulnerabilities of earlier solutions that relied on WebRTC gateways to mediate users' connections and which could enable attackers controlling the WebRTC gateway to detect the transmission of covert payload and inspect the content of the media streams. For instance, Protozoa [60], an earlier tool, was susceptible to certain types of man-in-the-middle attacks. Despite Stegozoa's significant improvements over past methods, it still does not achieve the high data transfer rates of systems like Protozoa, nor does it match the stronger security model of SkypeLine [61].

3.7 End User Network Systems

Two blocs of struggle against censorship that the majority of popular circumvention systems belong to are the end users and core network operation points [62]. Widely used circumvention systems such as Lantern and Tor [7], with more than two million daily users, are categorized as end-user network systems [63]. These proxies demand minimal resources to operate, making them accessible to a wide range of end users and promoting a potentially huge support base. However, regimes and nation-state censors are actively seeking to detect and block user proxies [62]. For example, censors can effortlessly gather Tor relay node addresses from their publicly advertised IP addresses. Even for private Tor bridges, whose addresses are handed out through unconventional channels such as email or moat, censors have demonstrated the ability to identify and effectively block them [64]. From the censor's standpoint, blocking individual proxy addresses causes minimal collateral damage in terms of economic or social disruption, as proxy addresses infrequently host services of significance to the censor.

3.8 Core Network Systems

Decoy routing [65], Domain fronting [66], and circumvention techniques operating on the Internet's critical and backbone infrastructure are generally categorized as Core network-based circumvention systems. Decoy Routing is accomplished by getting help from Internet Service Providers or Autonomous Systems (ASes) who own the routers in the middle of the network [67]. It identifies packets with embedded steganographic signals by secretly rerouting traffic from the network core to covert destinations by intercepting and screening through network traffic. While it is understandable that the host can easily achieve filtering and blocking IP addresses, blocking and filtering well-placed routers or other transit devices in the network is difficult.
In three points, Karlin et al. [65] define the Internet's current architecture properties that can be leveraged for Decoy Routing. At first, IP packets do not contain router addresses; therefore, it is challenging to filter routers at the IP level; secondly, A network does not have much control over the upstream paths that their packets take because IP routing is federated, and finally, it is difficult and unreliable to filter out all the paths that go through a specific router. Moreover, the router's paths can significantly impact how much of the Internet the adversary can reach, and a well-placed router can lie on the path of a client and a huge number of destination addresses. However, such systems necessitate collaboration from ISPs to operate, thereby presenting a significant obstacle to implementation and deployment [45].
In content delivery networks, Domain fronting and Domain shadowing harness the capabilities of content distribution networks to obfuscate covert domains within network requests and reroute them seamlessly within the CDN's infrastructure [66, 68]. These types of methods operate on the assumption that censors are hesitant to block vital infrastructures due to the significant unintended consequences involved. Blocking the core networks could result in considerable performance issues or complete service disruption. The obstruction against Domain fronting and similar techniques is that popular CDNs such as Google, Amazon, and Microsoft Azure have taken measures to prevent them on their platforms, preventing collateral damage that may happen as a

result of nationwide censorship [69].

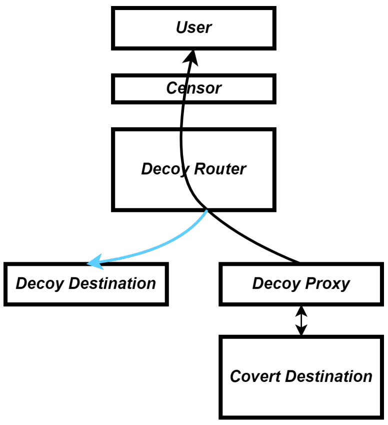


Figure 1: An overview of Decoy Routing

3.9 Edge Networks Systems

Kon et al. [62] argue that End-user networks and Core network fronts are not enough, and more operating points are needed to balance the huge power difference between the capable and powerful censors and the normal censored users. They want to open a new front against censorship by leveraging *Edge networks*. Campus networks, enterprises, and private data centers are considered edge networks. they possess advantageous characteristics due to their strategic position within the network architecture and deliver diverse services, including educational resources, software downloads, and web content. They also possess substantial resources, including publicly routed IP address blocks and sophisticated hardware equipment, such as computing and high-speed networking infrastructure. Kon et al. [62] want to deploy a third base of support by taking advantage of the mostly untapped edge networks-based circumvention approach by introducing their circumvention system, *NetShuffle*.

As its name implies, *NetShuffle* uses a proxy shuffling technique to circumvent censorship. This idea shares similarities with IP hopping, presented by Govil et al. [70], where users frequently change their IP address, protecting their identity from an adversary while initiating a flow and preventing linking multiple flows to a single client. *NetShuffle* achieves unblockability by disassociating services from their public identifiers through proxy shuffling. It mixes up the mapping between participating network domains and their IP addresses, detaching from the traditional method of locating proxies through fixed identifiers and preventing blocking.

3.10 Summary of Circumvention Techniques

The following table gathers the circumvention systems, techniques, and protocols reviewed in the previous sections. It includes a summary of each technique and its strengths and weaknesses. Some papers measure the effectiveness of a technique or a protocol; the summary, strengths, and weaknesses of those techniques and protocols are also included in the table.

Table 5: Summary of Circumvention Techniques.

Name	Summary	Strengths	Weaknesses
Infranet [43]	Enables users to circumvent by establishing covert channels with accessible web servers.	• Servers provide clients access to censored sites while continuing to host normal uncensored content. • It can carry out a remote login session.	• Infranet requester software was to be distributed on physical copies such as CD-ROMs. This type of distribution mechanism is slow, provides evidence for culpability, and is easy for censoring governments to control. • The architecture could not protect against an impersonation attack. The censor establishes an Infranet responder and discovers requesters by identifying the Web clients that send meaningful Infranet requests.

Decoy Routing [65]	Leverages the Internet's architecture, where router paths are not directly discernible in IP packets, routing is federated, and a strategically positioned router is used to achieve free Internet access.	• A strategically positioned router makes blocking difficult without disrupting substantial Internet access. • IP routing is federated, and the network does not have much control over the upstream paths that their packets take.	• Difficult to deploy; the system depends on the cooperation of ISPs, which poses a major barrier to their implementation and widespread use. • The difficulty in implementation and deployment and the need for outside collaboration make it a costly solution.
Browser-Based Proxies [44]	Creates millions of short-lived proxies, each proxy could be active for just a few minutes.	• The proxy disappears and leaves no trace on the visitor's machine when they navigate away from the website.	• If the attacker gets hold of the user's critical information, such as IP addresses and port numbers, transmission through proxies can be interrupted easily [71,72]. • The adversary could hijack the traffic and disclose the transmitted data if the proxies are contributed or manipulated [71,73].

VPNs [6, 59]	WireGuard uses state-of-the-art ChaCha20-Poly1305 authenticated encryption and encapsulates IP packets into UDP, and OpenVPN provides IP or Ethernet encapsulation into TCP or UDP protocol.	• They provide confidentiality, integrity, and authentication for encapsulated traffic. • OpenVPN can hide TLS negotiation from eavesdropping by using a pre-shared key that fully supports TLS handshake. • WireGuard has its handshake implementation. The client and server are authenticating others with their own public keys.	• VPNs can be easily blocked by Internet service providers when they are discovered.
ESNI [11]	It is a new extension for TLS 1.3 and has been deployed to most web browsers and more than one million Alexa top sites.	• Fixs and prevents server name and hostname leakage similar to the ones of TLS 1.2 and non-encrypted DNS traffic. • Along with encrypted DNS channels, it helps to unblock a number of websites currently censored in China.	• There have been reports from July 2020 of the GFW blocking all ESNI traffic [50]. Indicating blocking ease. • 1.5–2.25% of domains of TLD zone files have a valid ESNI key. Indicating the low adoption of ESNI [50].

uTLS [7]	A client TLS library that provides precise control over TLS handshakes.	• Enables tool maintainers to automatically mimic other popular TLS implementations. • Allow to mimic popular fingerprints by developers by automatically copying generated code from our website to configure uTLS.	• May not fully support Client Hello messages that were generated by automated fingerprint codes. • Connections can visibly break if servers select a cipher suite that is still not implemented by uTLS.
MultiProxy [53]	Developed utilizing a blockchain-based economic model as a Peer-to-Peer (P2P) circumvention system.	• Overcomes the shortcomings of the client-server-based architecture by combining the advantages of both client-server. • Provides a latency close to the latency of a direct server connection while providing good protection and privacy for the user.	• The system needs multiple traffic obfuscation methods to evade the GFW. • It needs a user-friendly and configurable GUI interface based on the existing command-line interface.

Geneva [55]	An automated tool that evolves packet-manipulation-based censorship evasion strategies using a genetic algorithm against nation-state-level censors.	• The ability to automatically discover censorship evasion strategies against on-path network censors. • The genetic building blocks of Geneva allow it to both re-derive all previously published supported schemes alongside deriving altogether new strategies that prior work was not posited as effective.	• Can fail to discover strategies only when its access to header fields is heavily restricted. • It is a relatively new technology that needs time for wider deployment.
MassBrowser [47]	A volunteer-run proxy-based system that leverages Domain fronting and CDN browsing. The proxies are not hosted on public IPs but rather hosted behind public NATs.	• Cheap cost of operation by leveraging shared IP addresses to defeat IP enumeration. • User can choose to tunnel some or all of their connections through the Tor interface of MassBrowser, giving them adjustable privacy. • Not as simple as blocking the IP addresses, as it would lead to the blockage of other clients behind the NATed IPs.	• The volunteer (non-censored) party can learn about the destinations accessed by their connected clients. • It can be insecure if the censors are intercepting the TLS communications.

Turbo Tunnel [58]	Uses a session/reliability layer in the middle of the protocol stack between the user's application streams and the obfuscated network connection responsible for evasion.	• The session/reliability layer is not exposed to the censor; it is an internal layer. P packets are encapsulated and transmitted inside a blocking-resistant obfuscated tunnel. • Permits switching between different obfuscation strategies without losing the end-to-end session state, providing more freedom and flexibility.	• It is an idea in its early stages; it is being presented as a general design pattern rather than a usable code or a reusable library. • Lack of built-in cryptography necessitates finding a suitable cryptographic protocol compatible with the Turbo Tunnel model.
Domain Fronting and Domain Shadowing [66, 68]	Utilizes Content Distribution Networks to camouflage covert domains within network requests, redirecting them through the CDN's network.	• Makes censors hesitant to block vital infrastructures; interfering with core network systems could significantly degrade performance or entirely halt services. • Easy to deploy, does not require network intermediaries' cooperation, and has been effectively used in systems like Tor, Lantern, and Psiphon.	• Action taken by major CDNs such as Google, Amazon, and Microsoft Azure, which have implemented policies to deter the use of their platforms for these techniques [69].

HTTP/3 over QUIC [19]	A new version of the HTTP protocol. It uses QUIC as the underlying encrypted transport.	• Provides reduced connection setup time and always-on, built-in encryption. • Provides a lower vulnerability to modification and tampering by middle-boxes in its connections.	• If an HTTPS request times out during a TCP handshake, the HTTP/3 request also fails before the QUIC handshake completion. • Can be blocked as UDP endpoint blocking use against HTTP/3 connections was detected [19]. • Blocklisting IP addresses is an issue for HTTP/3 requests; IP blocking prevents HTTP/3 requests to hosts who are blocklisted.
Encrypted DNS [25]	DNS traffic that is secured using protocols such as DNS-over-TLS (DoT) and DNS-over-HTTPS (DoH). Protocols that mitigate privacy concerns by encrypting the communication between clients and DNS resolvers.	• Manipulating encrypted DNS traffic is challenging for on-path censorship devices. encrypted DNS resolvers have significant potential to help censored end-users.	• The effectiveness varies from one country to another; it is not always possible to access censored domains. • It does not help with accessing domains that are affected by other types of censorship techniques, e.g., SNI-based blocking, IP-based blocking, and HTTP-based blocking.

BlindTLS [12]	Introduces minimal overhead by selectively transferring packets over an encrypted TCP proxy and hides the true SNI value in TLS 1.2.	• Improves the client's overall performance by sending most of the network's traffic directly to the server. • Demonstrated the ability to connect to 54% of blocked websites from a major Indian ISP.	• ISP could identify and block BlindTLS through active probing attacks. • ISP could prevent evasion by blocking communication to proxy nodes. Achieved by exploiting the remote port forwarding capabilities of SSH and VPN gateways. • Just by simply dropping all TLS session resumption packets, ISP could block BlindTLS.
Camoufler [45].	Utilizes Instant Messaging (IM) platform as a medium to tunnel the censored traffic to achieve minimal latency, adequate throughput, reliability, and blocking resistance.	• Reasonable adaptation and sufficient QoS for web browsing implemented on five popular IM applications, including Telegram, Signal, Slack, Skype, and Whatsapp, and with the feasibility of extension to others. • Low latency accessibility to Alexa top-1000 web pages with 4.1s and 3.6s, respectively, being the average median time.	• Adversary can identify and censor the IM IDs of Camoufler's servers. If an adversary is the owner of the involved IM platform, they can filter the IM IDs by themselves. • Evoking suspicion of using Camoufler by long-term user profiling by adversaries is a present issue.

Stegozoa [61]	It embeds the covert data steganographically into the WebRTC video signal. It uses steganography techniques and applies them within the WebRTC video coding pipeline.	• Protects against man-in-the-middle attacks, direct video content inspection, and detection of covert payloads. • Calls cannot be distinguished from a normal conference call even if a censor [74].	• Limited deployment as it was designed for experimental environment use and tested within the user's network, not for use alongside Internet censorship evasion tools and existing networks. • It still lacks the massive throughput of similar systems like Protozoa and the stronger threat model of SkypeLine [61].
Social Network Friendship Enhanced Decentralized System to Circumvent Censorships (SEnD) [46]	Built upon an overlay and leverages IPOP (IP-over-P2P) tunnels, enables users in an uncensored area to act as proxy servers for their social friends in a censored area.	• Tunnels traffic in a peer-to-peer fashion with encapsulated, encrypted messages without the need to pass through centralized gateways. • Leverages trusted social relationships to find and establish connections between supporters and requesters (users). • Prevents systematic prediction or probe by enabling traffic proxies to use dynamically allocated NATed endpoints.	• The reliability can be impacted by the behavior of individuals within the network and the availability of supporter nodes, which may lead to inconsistent performance. • A decentralized system integrated with IPOP tunnels and built upon an overlay where each user has a private IP channel to their social friends is a complex system and can lead to scalability issues.

4 Selected Techniques

This chapter provides an in-depth review of the chosen techniques that integrate into our proposed scheme. Following the investigation into the current state-of-the-art circumvention techniques, we formulated the concept of the proposed scheme. We selected three main techniques that can be combined to collectively achieve the scheme's goals and desired results. The chapter is outlined as follows: Section 4.1 presents the idea of *proxy shuffling* and provides detailed descriptions of the components and workflow of the *NetShuffle* [62] circumvention system. Section 4.2 presents the *Lox* proxy and bridge distribution system [75] and outlines its features, components, and functions. Section 4.3 outlines the concept of a probe-resistant proxy service, forming the foundation of our proxy service implementation.

4.1 Proxy Shuffling

The idea of shuffling IPs and constant changes in network location was introduced to censorship circumvention through various projects [76, 77]. However, few methods have concentrated on enhancing the resilience of proxies against blocking as significantly as NetShuffle.

4.1.1 NetShuffle

NetShuffle [62] is a censorship circumvention system designed specifically for edge networks [ii]. It introduces the *proxy shuffling* approach that aims to make proxy services more difficult to block by segregating them from their public identifiers by a random, fast, and continuous change of proxies. In contrast to earlier censorship-resistance methods that tie between fixed identifiers and their proxies, which can easily be exploited by blacklisting [78]. The shuffling algorithm is implemented within a programmable switch, which enables seamless integration with existing edge network infrastructure without requiring significant modifications. NetShuffle provides censored users with domain names that look harmless and serve as gateways to access blocked content. When a client queries the provided domain name, NetShuffle's DNS resolver provides a temporary, client-facing IP address, which is not the actual internal IP address. The border router translates incoming and outgoing connections to this client-facing IP address. The main benefits of NetShuffle include:

1. **Enhanced unblocking capability:** Shuffle proxies are significantly more difficult to block due to dynamic mapping and frequent IP address changes.

[ii]The concept of Edge Network is detailed in Chapter 3, specifically in Section 3.9

2. **Ease of deployment:** NetShuffle operates transparently to internal users and services, maintaining the existing network structure and using already available edge network resources.

3. **Scalability:** NetShuffle can handle many users and proxies without compromising performance.

4.1.2 The NetShuffle Algorithm for Shuffling

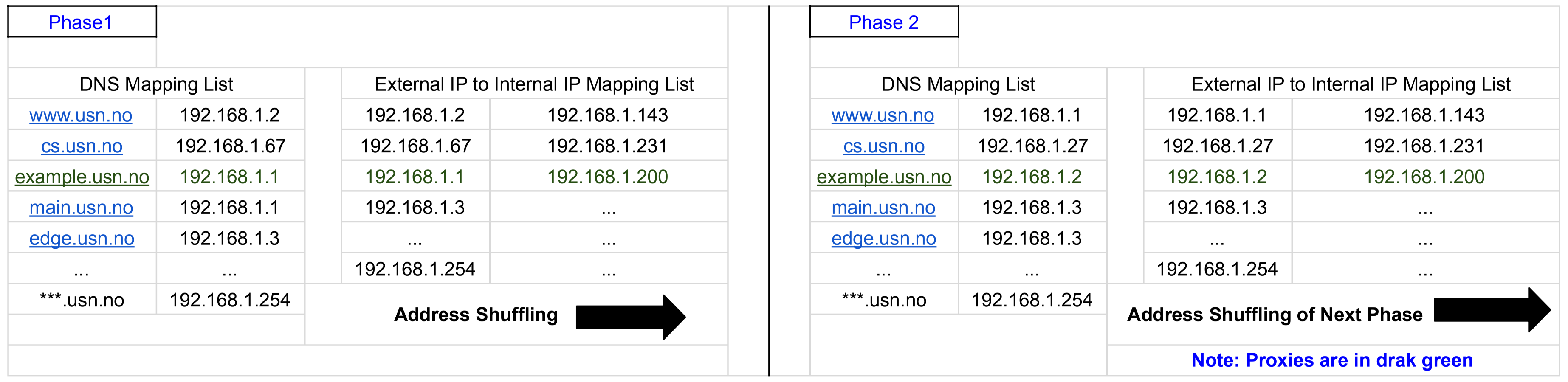
In the *NetShuffle* algorithm, a shuffle is a random reassignment of IPs. The *NetShuffle* algorithm differentiates between externally visible and internally used IPs by defining *ExternalIPs* as the publicly facing IP addresses exposed to clients [62]. In contrast, the true IP addresses, which are never affected by the shuffle, are represented by *InternalIPs* for edge services. The real *InternalIP* corresponded to a specific *ExternalIP* is revealed by using a function denoted as Π_{IP}.
The Border switch is a device that plays a crucial role in the shuffling process. It contains the mapping responsible for translating the destination IP addresses of incoming packets from *ExternalIPs* to *InternalIPs.* The internal addresses are protected and not visible to users because the function Π_{IP}^{-1} performs the opposite action of converting outgoing packets' source IPs from *InternalIPs* to *ExternalIPs.* To maintain consistency, each epoch generates new random permutations Π_{IP} and the reverse Π_{IP}^{-1}. Additionally, a DNS mapping Π_{DNS} is generated; it maps all advertised subdomain names to their current *ExternalIPs.* This mapping is installed on the authoritative DNS server to answer resolution requests.

4.1.3 The Flow of NetShuffle's Address Shuffling

This section details NetShuffle's address shuffling spanning two phases presented in Figure 2 and the process of employing the shuffle to confuse censors.

- **First Phase:** The first step of connecting a user to an internal IP starts at Phase 1, outlined in the four points below for easy understanding:

  1. The user is given a proxy linked to an external IP address; in this case, *example.usn.no.*
  2. The proxy *example.usn.no* translates to *192.168.1.1*, the corresponding external IP assigned to it in the DNS mapping table.
  3. The user connects to *192.168.1.1*, which has a true corresponding internal IP address in the external IP to the internal IP mapping list.
  4. The shuffling switch redirects the user's connection to the true internal IP at *192.168.1.200.*



Phase1

DNS Mapping List	
www.usn.no	192.168.1.2
cs.usn.no	192.168.1.67
example.usn.no	192.168.1.1
main.usn.no	192.168.1.1
edge.usn.no	192.168.1.3
...	...
***.usn.no	192.168.1.254

External IP to Internal IP Mapping List	
192.168.1.2	192.168.1.143
192.168.1.67	192.168.1.231
192.168.1.1	192.168.1.200
192.168.1.3	...
...	...
192.168.1.254	...

Address Shuffling

Phase 2

DNS Mapping List	
www.usn.no	192.168.1.1
cs.usn.no	192.168.1.27
example.usn.no	192.168.1.2
main.usn.no	192.168.1.3
edge.usn.no	192.168.1.3
...	...
***.usn.no	192.168.1.254

External IP to Internal IP Mapping List	
192.168.1.1	192.168.1.143
192.168.1.27	192.168.1.231
192.168.1.2	192.168.1.200
192.168.1.3	...
...	...
192.168.1.254	...

Address Shuffling of Next Phase

Note: Proxies are in drak green

Figure 2: NetShuffle's address shuffling spanning two phases

- **Second Phase:** The mappings of NetShuffle are randomly shuffled as the first phase passes, making it challenging for censors to block the external IP addresses linked to hosted proxies. The reason is that an external IP previously accessing a proxy might later direct to a legitimate service. For instance, the external IP *192.168.1.1*, initially connected with the hostnames *example.usn.no* and *main.usn.no*, is later rerouted to the valid *www.usn.no* service. Consequently, if a censor monitors and blocks the external IP resulting from the user's DNS query for *example.usn.no*, it probably also blocks a legitimate service in future phases, resulting in collateral damage. Similarly, an external IP formerly pointing to a genuine service might be reassigned to a proxy, as can be observed with external IP *192.168.1.2*, which shifted from *www.usn.no* to *example.usn.no*. Therefore, any attempt to create a list of approved external IP addresses for legitimate services is ineffective for the censor.

4.2 Proxy Distribution

Proxy and Bridge distribution systems are developed to safely and publicly distribute large pools of proxies and bridges to users in censored regions. Recent projects like the Lox Bridge and Proxy Distribution System were built to be enumerate-resistance and protect the client's privacy while being capable of efficiently working alongside other circumvention techniques.

4.2.1 Lox Proxy Distribution System

Lox is a system that provides an enumeration proof and safe proxy and bridge delivery process to users who wish to access the free Internet from their censor-controlled areas [75]. Lox aims to protect clients' privacy by a scheme called the unlinkable multi-show anonymous credential. This method allows users to prove their identity without revealing it, which suits scenarios with a single credential issuer and verifier. NetShuffle and similar systems need proxy distribution systems that provide anonymity and a high trust level to circumvention while smoothly distributing their proxies to clients and protecting them from widespread blockage from censors [79].

4.2.2 Issues Facing Distribution Systems

Generally, malicious actors try to enumerate proxy distribution systems by impersonating genuine users. This is achieved by entering the system with fake accounts called Sock-Puppet [75]. As a solution, *Lox* does not try to invent a completely new method for solving this problem; it rather combines features from earlier systems to propose a more robust method. *Lox* wishes to allow a larger number of untrusted users to join the system by prioritizing the client's social graph protection.

4.2.3 Lox's Features

Lox protects users and limits the impact of a censor's malicious behavior by integrating key features from earlier bridge distribution systems.

1. Lox integrates the ideas of Hyphae [80] and rBridge [81] for a credit scheme and reputation system to manage how users access and use bridges in the Lox [75] system. The users are assigned different trust levels. These levels determine a user's privileges or access rights within the system. Users must privately prove that a certain period has elapsed since they first learned about bridges that are still unblocked to increase trust level. The system can gauge whether the user is reliable and not working with censors to block bridges by requiring a period of time to pass.

2. Lox implements a bootstrapping period strategy to allow trusted users to establish groups to maintain bridge access. In this period, only trusted users can be invited to use the system, and this initial phase is essential to counteract the most effective strategies censors employ.

3. Lox integrates an anonymous suspicion as attributes credential scheme similar to the system used in Hyphae and a trust levels system similar to the one in Salmon [82]. The trust level attribute indicates how long a user has been in the system without their bridges being blocked, and suspicion increases when a trusted user's bridges are blocked, indicating potential risk.

4. The bridge distribution server in Lox is designed not to learn, store, or log identifiable or linkable information about users' social graphs. Lox aims to make it impractical, if not impossible, for a censor to identify users' friend groups, even if the bridge distribution server is compromised or its information is confiscated.

4.2.4 Lox's Components

We outline the core components of Lox as follows:

- **Lox Authority:** Lox operates with a central Lox Authority (LA), serving as the issuer and verifier of anonymous credentials and tokens across all Lox protocols. The LA maintains a bridge database, which includes numerous bridge records essential for connecting to the open internet.
- **Lox anonymous credential:** The cryptographic tool that enables the authentication of attribute collections without compromising user anonymity is considered Lox's credential.

Table 6: The descriptions of attributes and symbols used in this section.

Symbol	Description
L	Is a user trust level attribute assigned by the Lox Authority.
a	Is an invitation countdown counter attribute that allows users to issue invitations.
β	Is user's assigned bucket, Users receive *i* attribute as bucket ID and *Ki* attribute as the encryption key used to encrypt the bucket.
t	Is time attribute marking the time and days users last updated their trust level. When new users join, the *t* is marked by their joining date.

The system allows the LA to verify the authenticity of a credential without linking it back to the specific issuance. Users can confirm the authenticity of their credentials using the issuer's public key. Lox employs these credentials, designed to be spent once and issued and verified by the LA. The user's Lox credential contains several attributes detailed in Table 6.

4.2.5 Lox Authority's Interaction with Users

The user may have several interactions with the Lox Authority through the Lox protocols, which involve the following stages as presented in Figure 3. Trust level L is an attribute assigned by the LA when a user's credential is initially created. New open-entry users joining Lox without an invitation start with a trust level of $L = 0$, while those invited begin with $L = 1$. Users can increase their trust level by demonstrating to the LA that their bucket's bridges have remained unblocked for a specified duration. Maintaining a credential for a bucket without blockages allows users to progressively increase their trust level, submitting periodic requests to the LA up to a maximum level of $L = 4$. Additionally, the invitation countdown counter *a* is a feature enabling users to issue invitations after they have reached a trust level of $L \geq 2$. Users reaching $L = 4$ receive $a = 6$ and those had $L = 4$ for $t \geq 84$ days receive $a = 8$. With every invitation a user sends out, the value of *a* decreases by one. Learning about these attributes is important to understand how the user interacts with the Lox Authority. Table 6 presents the different attributes of the user's Lox credential. The user's different interacts with the Lox Authority are outlined as follow:

- **Open-entry and Credential Issuance:** Users can obtain a Lox credential by presenting an open-entry invitation, verified in zero-knowledge by the Lox Authority. This ensures the user's anonymity while allowing them access to the system. The LA sets the initial credential issued unilaterally, except for the credential ID, which is chosen jointly to prevent transaction linkage to the user. This mechanism allows for secure and private distribution of network access credentials.

- **Trust promotion and migration:** Trust promotion and migration in the Lox system facilitate users' transition from a non-trusted status (L = 0) to a trusted one (L = 1) and grant them access to a privileged *SuperBucket* credential featuring three bridges, subsequently ceasing to distribute bucket's credential to new, non-trusted users. Trust promotion is triggered when a bridge remains reachable for 30 days. At this point, users with access to that bridge can upgrade their status and obtain credentials for a more trusted network segment. This cessation is timed just before the initial set of non-trusted users who had access to the bucket qualify for elevation to trust level $L = 1$. Following this, disseminating the trusted bucket credential to new users solely occurs through invitations from existing bucket users. The process involves verifying user credentials and issuing new credentials without revealing the user's previous or new bridge connections
- **Invitation Protocol:** Users who have reached a trust level of L = 2 are authorized to send out invitations. These invitations elevate newcomers to a trust level of L = 1, granting them access to information of the same bucket (denoted by β) and user blockage data (denoted by d) of their inviter. Placing invitees in the same bucket as their inviter encourages the careful distribution of invitations to prevent abuse by potentially malicious actors from exploiting the system to discover more bridges. To send an invitation, users must submit their Lox credentials and bridge reachability certificate to the Lox Authority. The LA should verify these credentials without learning their contents (zero-knowledge), sign the new Lox credential and the new invitation credentials after the verification, and issue them to the user.
- **Blockage Migration Protocol:** When a bucket is considered blocked, the system recognizes that users with that bucket face limited options for regaining access to the system at their previous trust level. Upon encountering a blocked bucket, most users are presumed to need to re-enter the system as untrusted users. This can happen through open-entry invitations or invitations from friends affiliated with different (unblocked) buckets. However, users with a trust level of L ≥ 3 can migrate to a new, unblocked trusted bucket. As trusted users migrate, their d attribute increments by one. Upon the d attribute reaching a value of 4, users cannot elevate their trust level enough to qualify for further migration. Users can move to a different trusted bucket by invoking the check blockage protocol. This involves securely submitting their Lox credential to the LA without revealing its contents. After the LA verifies the credential's authenticity through a zero-knowledge proof, it provides the user with a migration key credential. The user then uses this key to decrypt their migration token and submits it to the LA along with their intended new Lox credentials as part of the migration protocol. Upon successful zero-knowledge verification of these credentials by the LA, it signs and issues the new Lox credential to the user. This new credential grants users access to the

new trusted bridges.

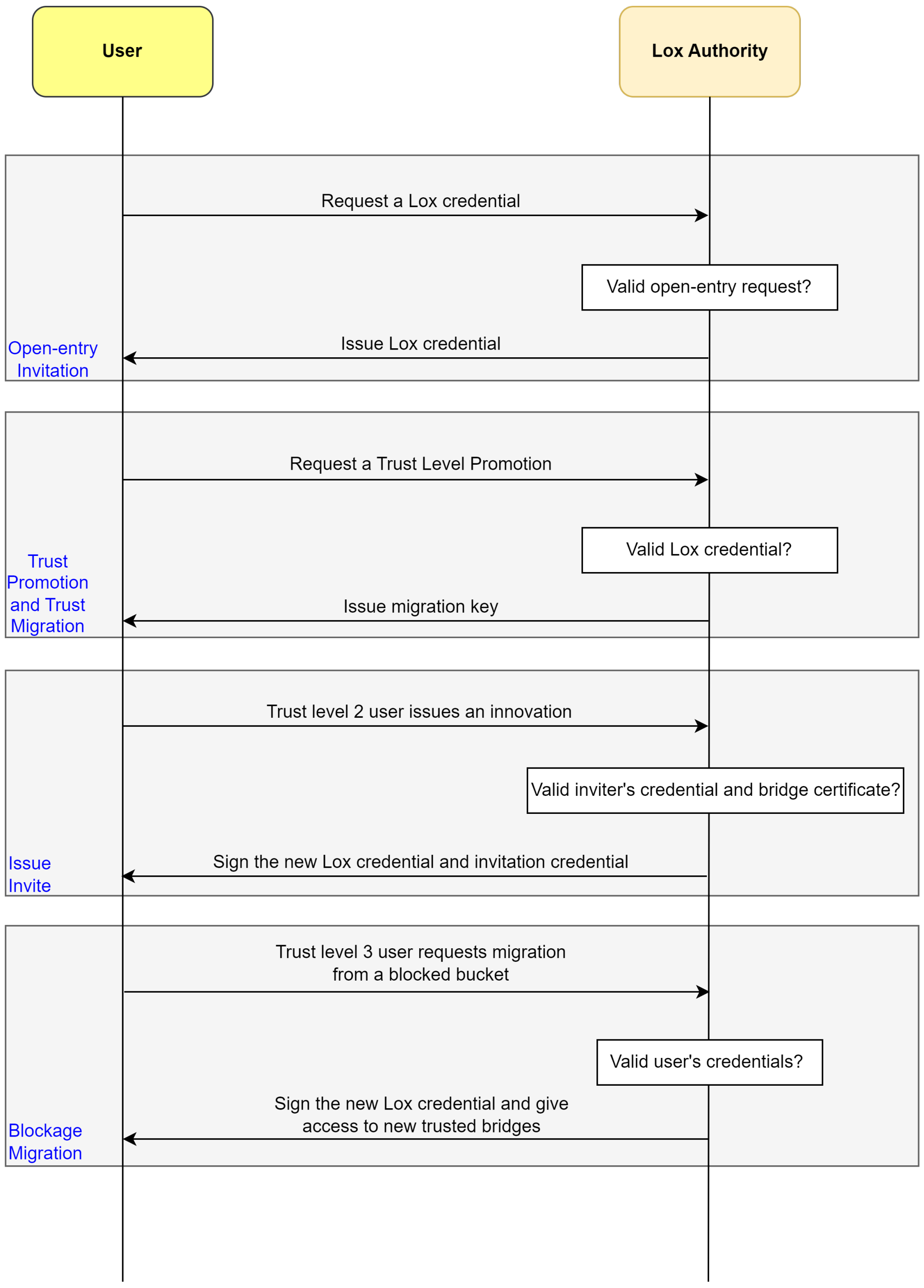

User
Lox Authority
Request a Lox credential
Valid open-entry request?
Issue Lox credential
Open-entry Invitation
Request a Trust Level Promotion
Valid Lox credential?
Issue migration key
Trust Promotion and Trust Migration
Trust level 2 user issues an innovation
Valid inviter's credential and bridge certificate?
Sign the new Lox credential and invitation credential
Issue Invite
Trust level 3 user requests migration from a blocked bucket
Valid user's credentials?
Sign the new Lox credential and give access to new trusted bridges
Blockage Migration

Figure 3: Lox Authority's interaction with users.

Table 7: Comparison between Lox and other similar bridge distribution systems.

Category \ Features	Salmon [82]	Hyphae [80]	rBridge [81]	Lox [75]
Multiple user participation models	✗	✗	✗	✓
Not store or log information about users' social graphs	✗	✓	✓	✓
Not reliant on third-party verification methods	✗	✓	✓	✓
Accommodating large user-base	✓	✗	✗	✓
Rewarding users with trust level	✓	✗	✗	✓
Dealing with invitees from blocked users with suspicion	✓	✗	✗	✓

4.3 Probe-resistant Proxy

While regular proxies and some probe-resistant proxies can generally route client traffic successfully and safely, there is still the possibility of proxy detection by censors. A proxy should provide active probe-resistant capabilities while avoiding looking like any normal protocol to avoid censor attention.

4.3.1 Challenges Encountered by Circumvention Proxy Systems

A study by Alice et al. [83] has found that the GFW identifies circumvention proxy systems by utilizing the incorporation of active probing attacks and passive traffic analysis. Examining the traffic characteristics and focusing on the entropy and size of the first data packet in each connection helps the censor identify traffic. Once a server is flagged, the server starts active probing and sends a series of active probes to the suspected server. These probes are not arbitrary; they consist of partial replays of past legitimate connections and random probes of various lengths. This aims to mimic legitimate traffic patterns or to elicit responses that would only come from the intended circumvention system's server, thus confirming its identity.

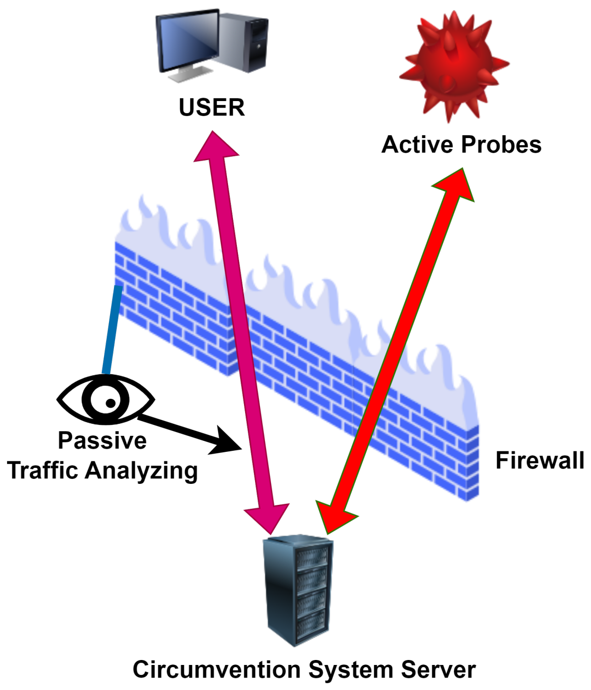


Figure 4: Passive Traffic Analysis and Active Probing Attack.

4.3.2 Designing a Probe-Resistant Proxy

We implemented a proxy design intended to resist active probing attacks. Our proxy utilizes the camouflage capabilities of HTTPS and the encryption techniques of the latest TLS protocols [79]. Active probing attacks are sophisticated techniques censors use to detect and block access to proxy servers. These attacks involve sending requests (active probes) to servers and analyzing their responses to identify characteristics that reveal them as the intended circumvention proxies [83]. The proxy probe-resistant should send plausible responses to censors to avoid detection. We can also leverage HTTPS's ubiquity and inherent security features to blend in with regular Internet traffic, making it significantly harder for censors to identify and block the proxy. The implementation chapter provides a detailed explanation of the features and capabilities of our probe-resistant proxy.

5 Proposed Scheme (ProxyPro)

We propose *ProxyPro*, an approach to circumventing Internet censorship by integrating state-of-the-art techniques to ensure client's safety and privacy in their censored region. This approach combines the user's social graphs protection and enumeration resistance of the Lox proxy distribution system [75], the probe resistance and disguising abilities of our probe-resistant proxy implementation, and NetShuffle's [62] constant proxy shuffling strategy to achieve a system that is resistant to replay attacks, enumeration, and blocking while maintaining focus on simplicity, ease of implementation, and cost-effectiveness. In this section, we present the core components that work together to run the different functionalities of the system. We identify the architecture and overall workflow of the system, which are supported by sequence diagrams and workflow charts.

5.1 Design Goals

ProxyPro aims to fulfill the several key design goals outlined below:

- **Enumeration Proof:** The proxy distribution process should be protected against enumeration attacks, and proxies in the proxy pool need to be protected from discovery by censors. By integrating the Lox proxy distribution system and its trust level scheme, ProxyPro aims to prevent large-scale enumeration attacks.
- **Probe Resistant:** The proxies used to connect users to free Internet must be developed and configured carefully to be protected from active probe attacks. ProxyPro aims to utilize the advantages of HTTPS and TLS data tunneling protocol integrated into our probe-resistant proxy implementation to achieve an overall better probe-resistant system.
- **Unblockability:** ProxyPro aims to archive a system that is significantly more robust and difficult to block by adapting the proxy shuffling strategy and dissociation of proxies from their address by the fast flux and the continuous change in the network.
- **High Quality of Service:** ProxyPro aims to utilize the capabilities of the edge network that provide a robust support base, e.g., a campus network. They provide high-speed networking and advanced computing infrastructure, translating to good QoS and long-term user advantages.

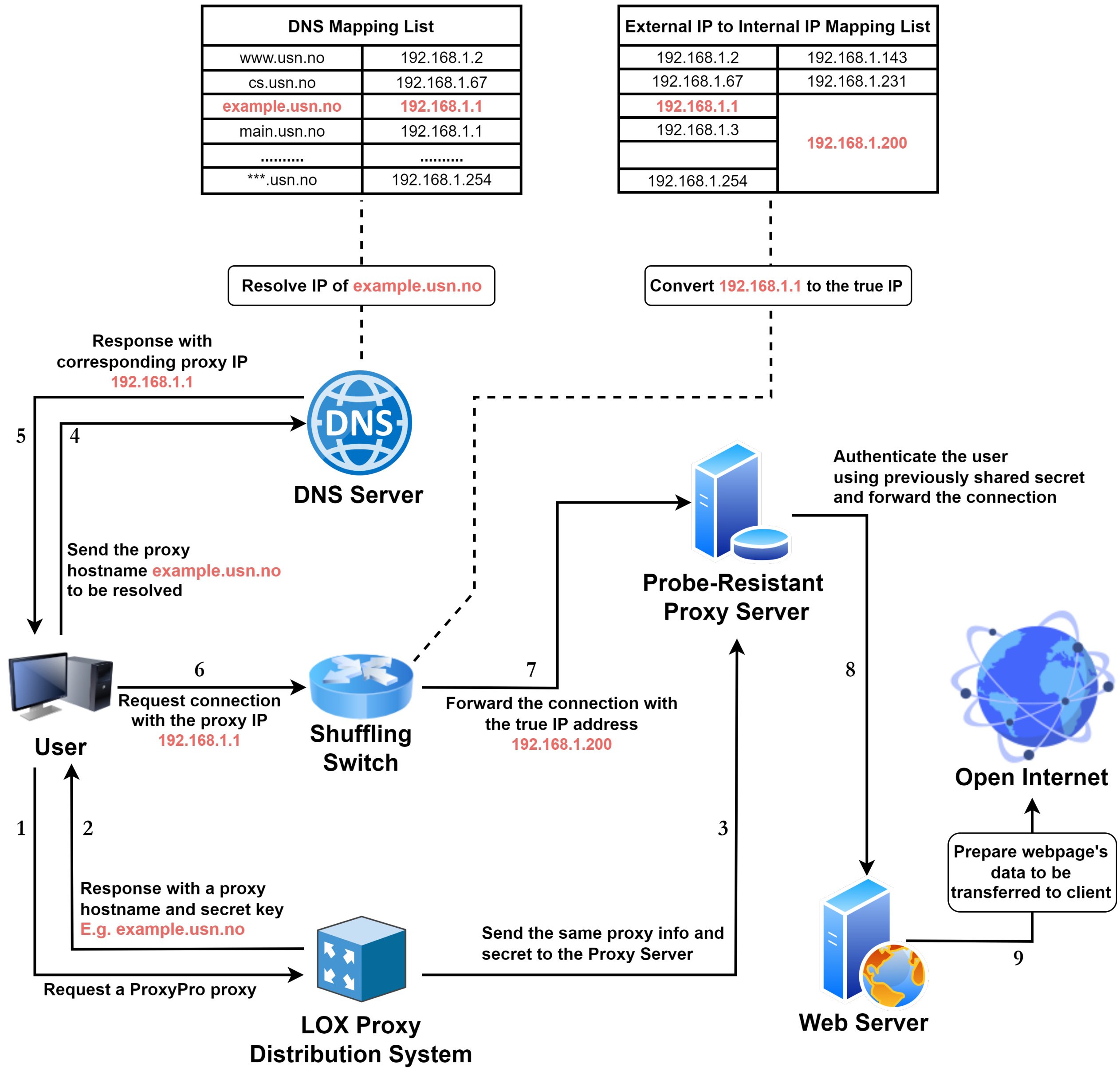
Table 8: Comparison between ProxyPro and several major circumvention systems.

Systems / Category	Domain Fronting [66]3.8	Decoy Routing [65]3.8	Tunneling [47]	Proxy-Based [47]3.1	ProxyPro
Deployability	○	●	◐	○	●
Low Cost	○	◐	◐	◐	●
Good QoS	●	●	○	○	◐
Difficult to block	◐	◐	◐	○	◐

5.2 ProxyPro's Components

We outline the core components of ProxyPro optimally working together to connect clients to free Internet as follows:

- **Proxy Distribution System:** The proxy distribution system acts as a proxy hostname and shared secret provider to the clients. The Lox proxy distribution system is the clients' first contact component to start a new connection. Clients access Lox and request a shuffling proxy.
- **DNS Server:** The ProxyPro's DNS resolver is the device that holds the DNS mapping table responsible for translating the client's query for the proxy hostname they received from the Lox proxy distribution system to the proxy IP address that they later use to request connection from the shuffling switch.
- **Suffling Switch:** The shuffling switch is the device configured with NetShuflle's algorithm for proxy shuffling. The algorithm is agnostic to the type of proxy used and shuffles any proxy provided to it. The switch also holds the mapping table required to translate the external proxy IP address that the client received from the DNS server to the true internal IP address.
- **Proxy Server:** The proxy server is the device that holds the proxy services and acts as an intermediary between the client and the web server. In our application, the proxy server can be configured with the probe-resistant proxy implementation and is the service that clients' traffic is forwarded toward by the shuffling switch.



DNS Mapping List	
www.usn.no	192.168.1.2
cs.usn.no	192.168.1.67
example.usn.no	192.168.1.1
main.usn.no	192.168.1.1
....	....
***.usn.no	192.168.1.254

External IP to Internal IP Mapping List	
192.168.1.2	192.168.1.143
192.168.1.67	192.168.1.231
192.168.1.1	192.168.1.200
192.168.1.3	
192.168.1.254	

Resolve IP of example.usn.no
Convert 192.168.1.1 to the true IP
Response with corresponding proxy IP 192.168.1.1
5
4
DNS Server
Send the proxy hostname example.usn.no to be resolved
Authenticate the user using previously shared secret and forward the connection
Probe-Resistant Proxy Server
6
Request connection with the proxy IP 192.168.1.1
User
Shuffling Switch
7
Forward the connection with the true IP address 192.168.1.200
8
Open Internet
1
2
3
Response with a proxy hostname and secret key E.g. example.usn.no
Prepare webpage's data to be transferred to client
Send the same proxy info and secret to the Proxy Server
9
Request a ProxyPro proxy
LOX Proxy Distribution System
Web Server

Figure 5: User's connection process via ProxyPro.

5.3 ProxyPro's Workflow

Figure 5 depicts the design of ProxyPro and the stages of connecting a user. Figures 6 and 7 provide a more detailed sequence diagram of the connection workflow. Additionally, Figure 8 presents a combined view of both sequence diagrams for easier understanding. Connecting clients to unrestricted Internet access in their restricted region via ProxyPro involves the following stages:

1. The client starts by accessing the Lox proxy distribution system and requesting proxy hostname information.

2. The Lox proxy distribution system responds to the request by returning a proxy hostname and a shared secret to the client. We use *example.usn.no* as an example for this demonstration.

3. The Lox proxy distribution system sends the same proxy information and a shared secret to the probe-resistant proxy server for later client verification.

4. The client sends the hostname *example.usn.no* to the DNS server to be resolved.

5. The DNS server compares *example.usn.no* with the IPs in its internal mapping table to find the corresponding proxy IP and returns it to the client. In this case *192.168.1.1*

6. The client then uses retrieved proxy IP *192.168.1.1* to send a connection request via the shuffling switch.

7. The shuffling switch then uses its external to internal IP mapping table to find the corresponding true proxy IP and then forwards the client's request to the probe-resistant proxy server.

8. The proxy server authenticates the client's connection using the earlier shared secret and safely forwards the request to the public web server.

9. The web server sends the requested webpage data back to the proxy server, which is then progressively forwarded back to the client via the relevant components.

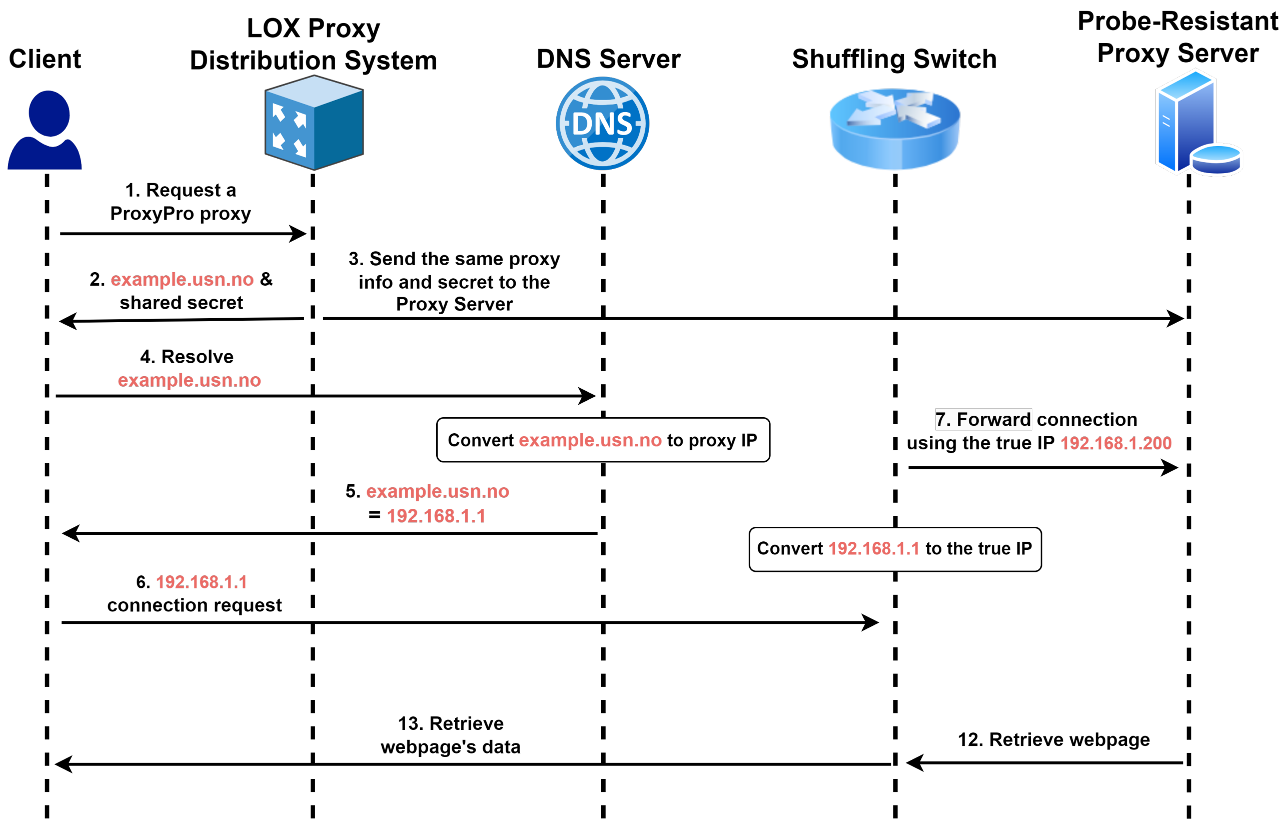

Client
LOX Proxy Distribution System
DNS Server
Shuffling Switch
Probe-Resistant Proxy Server
1. Request a ProxyPro proxy
2. example.usn.no & shared secret
3. Send the same proxy info and secret to the Proxy Server
4. Resolve example.usn.no
Convert example.usn.no to proxy IP
7. Forward connection using the true IP 192.168.1.200
5. example.usn.no = 192.168.1.1
Convert 192.168.1.1 to the true IP
6. 192.168.1.1 connection request
13. Retrieve webpage's data
12. Retrieve webpage

Figure 6: Sequence diagram depicting the interaction between Client, Lox, DNS server, and the Shuffling Switch.

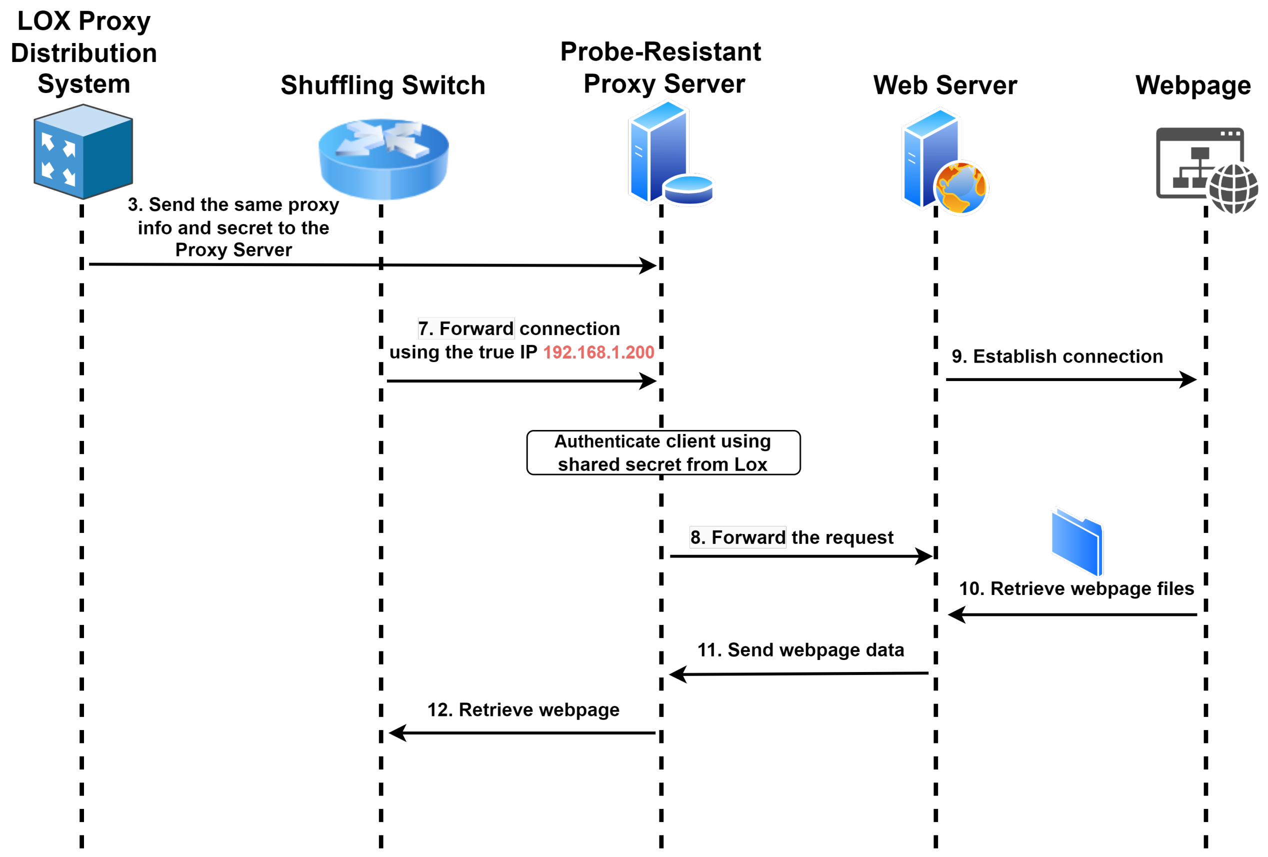
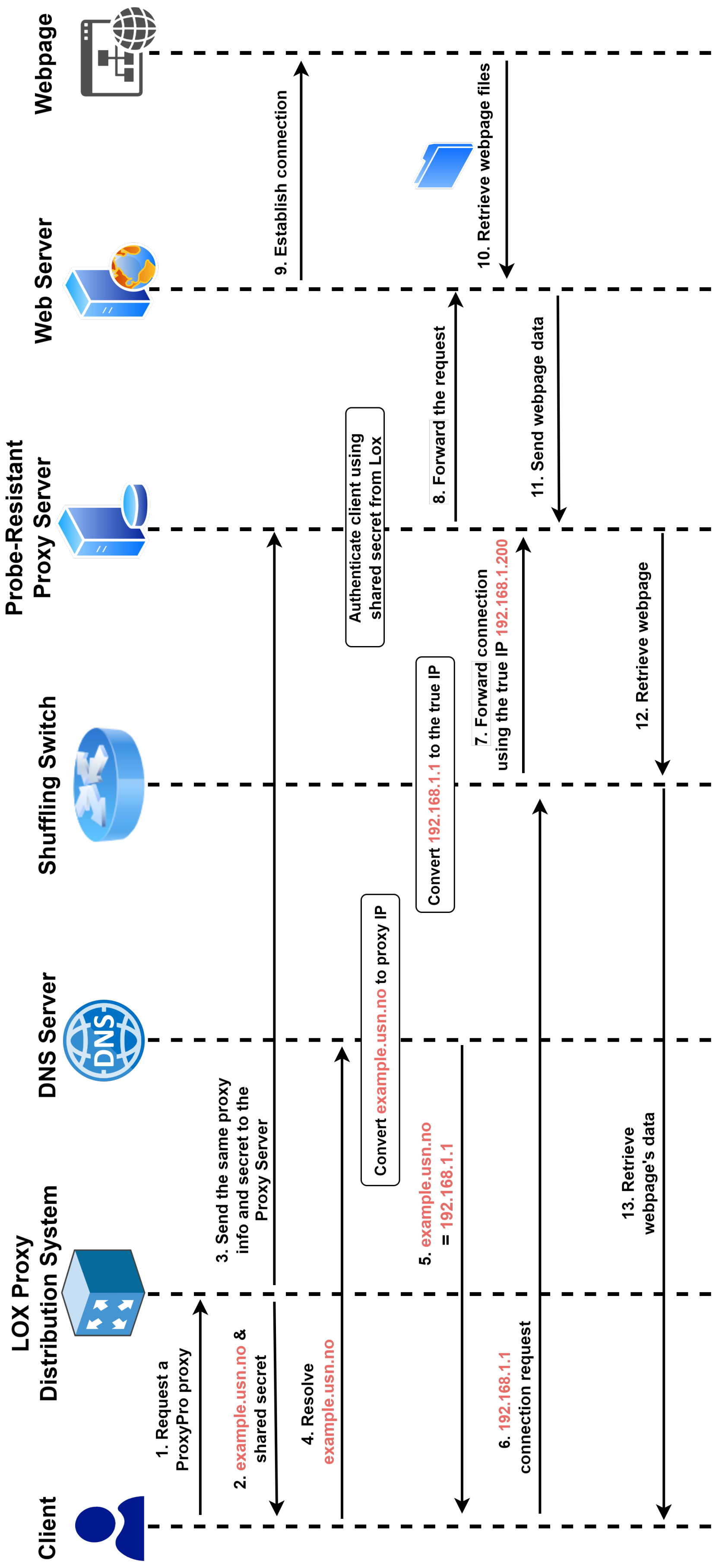

LOX Proxy Distribution System
Shuffling Switch
Probe-Resistant Proxy Server
Web Server
Webpage
3. Send the same proxy info and secret to the Proxy Server
7. Forward connection using the true IP 192.168.1.200
9. Establish connection
Authenticate client using shared secret from Lox
8. Forward the request
10. Retrieve webpage files
11. Send webpage data
12. Retrieve webpage

Figure 7: Sequence diagram depicting the interaction between Lox, Shuffling Switch, Proxy Server, and Web Server.



Client
LOX Proxy Distribution System
DNS Server
Shuffling Switch
Probe-Resistant Proxy Server
Web Server
Webpage
1. Request a ProxyPro proxy
2. example.usn.no & shared secret
3. Send the same proxy info and secret to the Proxy Server
4. Resolve example.usn.no
Convert example.usn.no to proxy IP
Authenticate client using shared secret from Lox
5. example.usn.no = 192.168.1.1
Convert 192.168.1.1 to the true IP
8. Forward the request
9. Establish connection
10. Retrieve webpage files
7. Forward connection using the true IP 192.168.1.200
6. 192.168.1.1 connection request
11. Send webpage data
12. Retrieve webpage
13. Retrieve webpage's data

Figure 8: Sequence diagram depicting the complete user and ProxyPro interaction.

5.4 Challenges and Limitations

The combination of several methods can usually lead to challenges in dealing with some weaknesses of the integrated systems. Efforts must be made to mitigate any adverse effects as much as possible. Below are some challenges that could impact our system's security, quality of service, and usability. Security, quality of service, and usability are three correlated aspects that influence one another in any system. We reference our design's security, functionality, and usability triangle in Figure 9 while addressing some challenges and limitations.

- The system's reliance on a trust level scheme and invitation-only access adapted from Lox can add complexity and potentially hinder usability for less technically capable users. The need to gradually build trust may deter new users who need immediate access to bridges, especially in urgent situations where quick and easy access to uncensored Internet is critical. This limitation is apparent in the design's security, functionality, and usability triangle in Figure 9 and Table 8, depicting a relatively high sacrifice of usability to gain higher security levels.
- Even with utmost optimization, running the shuffling algorithm and maintaining the necessary mappings and transformations within the network hardware could utilize significant computational resources. This might impact other operations, especially if we were forced to deploy in environments where resources are limited.
- Our probe-resistant proxy service shows higher latency compared to a direct connection establishment or systems like Shadowsocks. The proxy configurations using TLS 1.2 and TLS 1.3 have additional round trips compared to systems like Shadowsocks [79]. Specifically, TLS 1.2 has two additional round trips, and TLS 1.3 has one. While slightly slower, the performance results are comparable to a direct connection. We believe this minor compromise in speed is acceptable for our probe-resistant proxy service's enhanced security and resistance features.

6 Security Analysis

This chapter explores a range of potential attacks and vulnerabilities that could impact our design and mitigation strategies to counter these threats.

6.1 Sock-Puppets Attack

Sock-puppets are fake accounts created to mimic legitimate users that censors or attackers often use to disrupt services [75]. Socks-puppet accounts can significantly undermine proxy and bridge distribution systems connections by impersonating legitimate users. This allows attackers to gather information about these bridges, potentially blocking or restricting access.
We integrated a proxy distribution system into our design and want to protect it against sock-puppet attacks by using its enumeration-resistant capabilities. This allows us to protect against sock-puppet attacks while prioritizing user privacy and social graph protection. This involves the following several key mechanisms [75]:

- **Anonymous Credential System:** The system utilizes a keyed algebraic MAC anonymous credential scheme. This system ensures that credentials cannot be linked to specific users across multiple uses, which is vital for maintaining anonymity and preventing tracking by malicious entities. Users receive credentials that allow them to access bridges, but these credentials do not expose their identity.
- **Trust Level Scheme:** The system incorporates a formal trust level scheme that progresses as users demonstrate trustworthy behavior over time. Users start at a base trust level and can gain higher levels by proving their reliability over time. Higher trust levels are associated with greater access to system resources, such as a broader range of bridges. This progression system deters sock puppets since gaining significant access requires sustained and genuine usage, which is challenging for sock puppets to mimic without detection.
- **Social Graph Protection:** The proxy distribution system prioritizes the protection of the user's social graph, opting for mechanisms that allow users to gain access through existing social trust networks. The system leverages trusted user networks for distributing bridges, reducing the likelihood of sock-puppet infiltration as trust must be vouched for by existing, reliable members of the community.
- **Use of Invitations and Reputation Systems:** Access to the system, especially at higher trust levels, can be through invitations from already trusted users. This approach uses the social trust graph to prevent easy access by sock puppets. Additionally, reputation systems evaluate users' behavior and restrict their abilities based on their proven trustworthiness, which

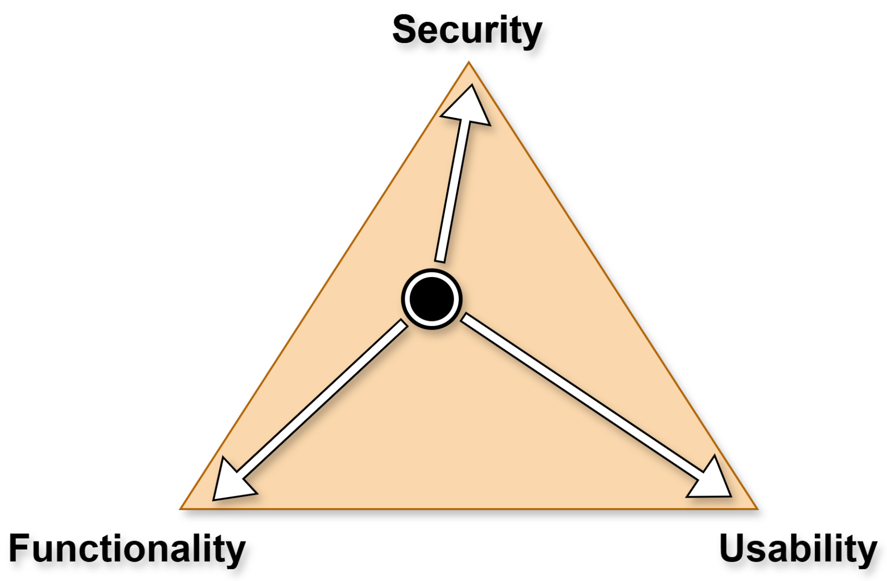


Figure 9: The Security, Functionality and Usability Triangle.

deters malicious activity.

- **Separation and Unlinkability of Open-entry Tokens:** Open-entry tokens used for initial access are designed to be unlinkable from further interactions within the system. This separation ensures that gaining initial access does not automatically grant further privileges, requiring users to build trust through continued and legitimate system use.
- **Invitation-only Access for Trusted Users:** Users considered trusted can be invited without needing to verify a valid third-party account, simplifying the process for genuine users and further securing the system against sock-puppet risks. This method allows the system to control access effectively without over-complicating the entry requirements.

6.2 Denial-of-Service Attack

This attack involves an adversary sending a large number of packets for compacted High Performance (HP) services without completing a TCP connection handshake, which can cause excessive buffering at the switch control plane [5]. In the context of our system, the controller plays a crucial role in processing TCP connection requests for compacted HP services. For these services, the controller must buffer the initial part of the connection before determining how to route them properly. This buffering is necessary because the service needs to read the initial packet data to make routing decisions based on the content.

An attacker could exploit this setup by initiating many connections to these HP services without completing the TCP handshake [5]. Since each initial packet needs to be buffered by the controller

to await possible completion and further processing, an excessive number of such incomplete connection attempts could overwhelm the controller's buffer capacity. This can lead to degraded performance or denial of service as legitimate traffic can no longer be processed effectively due to resource exhaustion.
We can mitigate this attempt of censors to flood the controller buffer by adopting the following techniques [62]:

- **Monitoring Access Frequency:** The system tracks how often a source IP address accesses the controller buffer. If this frequency exceeds a certain threshold, it implies potential malicious activity.
- **Connection Reset and Buffer Dropping:** When suspicious activity is detected from an IP address, the system can reset the ongoing connection attempts from that IP and drop all associated buffers. This helps free up resources in the controller.
- **Rate Limiting:** Applying rate limits to the number of new connections an IP can initiate over a time period prevents a single source from flooding the buffer.
- **Compaction Changes:** As a more dynamic approach, the controller can change the handling of the HP services not to require mediation by the controller for new epochs. This adjustment means that subsequent packets from previously flagged IPs would not need to be buffered for TCP handshaking, reducing the potential impact on the controller.

6.3 Resource Exhaustion Attack

Another type of attack that can try to exhaust our system is flooding the connection's table. Here, an attacker deliberately creates many connections to a network service or device but never completes them. This kind of attack specifically targets the connection tables of network devices like routers, firewalls, or, in the case of our system, the programmable switch [62]. These connection tables are used to keep track of active connections, including their states and other necessary details. When the attack happens, because the connections are never completed, the entries remain in the table indefinitely or until they time out, which can take a while. If the table fills up, it can no longer accept new connections, effectively denying service to legitimate users. As outlined below, we can follow several strategies to protect against connection table flooding.

- **Inactive Connection Monitoring:** The system can actively monitor the connection tables for incomplete entries. If a connection does not change state within a certain period, it is considered inactive or potentially malicious.
- **Blocklisting:** IP addresses identified as sources of multiple inactive connections can be blocklisted. Any further connection attempts from these addresses can be immediately

dropped or ignored, preventing them from consuming resources in the connection table.

- **Hardware Idle-Timeouts:** The system can automatically configure hardware-level timeouts to remove old or inactive entries from the connection table. This feature ensures that the connection tables are not filled with stale or malicious entries, maintaining availability for new, legitimate connections.
- **Control Plane Management:** The control plane, which typically manages the configuration and overall operation of the switch, can intervene to manage connection tables dynamically. It can adjust settings or clear entries based on current network conditions and threats.

6.4 Subdomain Blocking

Subdomain blocking is a technique censors or network administrators use to block access to specific parts of a domain by targeting its subdomains at the DNS level [5]. Censors might try registering as clients and obtaining proxy identifiers, making it difficult to build an effective blocklist due to the large space of subdomain names [62]. The adapted proxy shuffling and the dynamic subdomain generation capabilities can render subdomain blocking ineffective even if an enumeration attack becomes possible. The following bullet points explain the reasons:

- **Dynamic Subdomain Generation:** The system can continuously generate new service subdomains. This means that even if a censor blocks one subdomain, new ones not yet on the blocklist appear. It's a moving target that is hard to neutralize completely.
- **Large Subdomain Space:** The potential number of subdomains that can be generated is vast. The system can programmatically create many subdomains, making it impractical for censors to block all possible subdomains without significant effort and resources.
- **Collateral Damage:** Blocking subdomains indiscriminately can lead to unintended consequences. For example, blocking a subdomain that appears malicious but is part of a larger legitimate service could disrupt normal internet operations and access for regular users. This is particularly problematic if the main domain hosts multiple critical services.
- **Detection Avoidance:** Subdomain blocking typically requires the censor to have knowledge of which subdomains are being used for circumvention. The shuffling strategy of frequently changing subdomains means that when a censor identifies a subdomain to block, others may have replaced it, making the blocking action outdated.

Practical Example: In practice, if our system uses a domain like *example.usn.no*, it can create multiple subdomains such as *example1.usn.no*, *example2.usn.no*, etc., at random or programmed intervals. If a censor decides to block *example1.usn.no*, the system can generate a new subdomain like *example3.usn.no* to continue its operations. The continuous change in subdomains and their potentially high number complicates effective censorship without affecting legitimate traffic.

6.5 Replay Attacks

Replay attacks involve an adversary re-sending previously captured network traffic to a server to trick it into repeating an action, such as authenticating a session. This can be done by capturing network data and then retransmitting it to trigger an unintended effect or gain unauthorized access to a system [79]. The probe-resistant proxy system we developed has several mechanisms to protect against replay attacks:

- **TLS Handshake Nonces:** It protects against replay attacks by using the TLS protocol, which inherently includes mechanisms to thwart such attacks. Specifically, it leverages bidirectional nonces during the TLS handshake process. These nonces ensure that each session between a client and server is unique, thereby preventing an attacker from successfully replaying previously captured traffic to impersonate a legitimate user or session [7]. During the TLS handshake, the client and server contribute random data used in the key generation processes for that session. This means that even if an attacker captures a valid handshake, they cannot simply replay it to establish a new session because the cryptographic keys derived from the nonces would differ each time, requiring new and valid nonces to generate the correct keys for session encryption.
- **Session-specific Data:** Each session includes unique, session-specific data that cannot be reused in different sessions or replicated in replay attacks. This makes any repeated or delayed transmission immediately suspect and identifiable as a potential attack.
- **Encrypted Connections:** Encrypted connections ensure that the data details, including any nonces or keys, are secure from interception and tampering. This encryption adds a layer of security, making it even more difficult for attackers to execute a successful replay attack.

6.6 Full network environment blockage

This is a drastic approach by censors to block access to an entire network environment. This method involves blocking all or a significant portion of the IP addresses and domains associated with a network to prevent access to specific services. This kind of blocking can be considered when more straightforward methods fail to curb undesired activities effectively [5]. We aim to deploy ProxyPro on a campus network or a similar environment, referred to as edge networks. Edge networks, similar to core networks, contain valuable domains, services, and IP ranges that are crucial not only to the region employing censorship but also to its users. Historically, censors prefer more precise, fine-grained methods such as keyword or targeted DNS and IP blocking to minimize disruptions and avoid economic consequences. While broader blocking does occur, such as during sensitive political events, these are usually temporary and aimed at controlling specific situations rather than routine censorship practices. Moreover, the interconnected nature of the Internet means that actions taken by one country can have unintended spillover effects on others, deterring such drastic measures due to potential international repercussions [5, 62]. This diversity in censorship approaches among different nations further complicate decisions about blocking entire networks, as what might be acceptable in one country could be opposed in another to avoid the ramifications of over-blocking.

Despite the challenges and rarity of full network environment blocking, the possibility of such actions cannot be dismissed entirely. Therefore, we acknowledge that ProxyPro, or any other circumvention system, cannot be deemed entirely blocking-resistant. Given the slight possibility of such measures being implemented, it underscores that absolute security against blocking cannot be guaranteed, as depicted in Table 8 and given the slightly degraded security shown in Figure 9.

7 Implementation

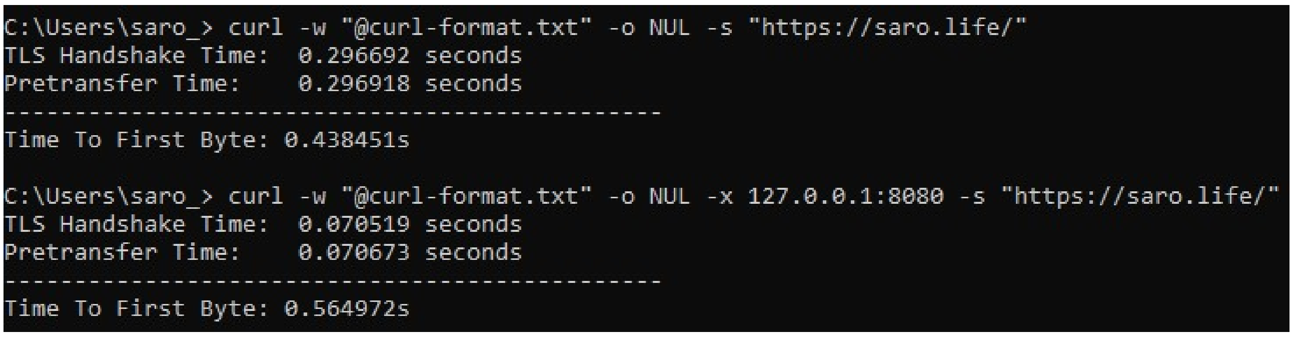
ProxyPro does not have a full implementation at this time. The allotted time of the thesis permitted only the design phase and a partial implementation for a project of such scale. The complete implementation of our proposed method can be a Ph.D. thesis spanning several phases or a collaborative team project featuring the researchers and developers who proposed and developed the various techniques integrated into our approach.
Nevertheless, we developed a probe-resistant proxy, an essential component of the ProxyPro scheme. In future development, this proxy can be seamlessly integrated into the overall ProxyPro system to serve its intended purpose, as detailed in the design chapter. Additionally, it can be utilized independently as an intermediary tool to enhance security and privacy, functioning like any standard proxy. We developed the proxy in Python and made it available on GitHub [iii].

7.1 Integrated Features and Protocols

This section details the features integrated into our proxy implementation to achieve its probe-resistant, security, and privacy capabilities. Additionally, it outlines the tests we conducted on the system and the analysis of the results.

- **TLS Integration:** We have mandated the use of TLS in our proxy system. The reliance on TLS protocol for data tunneling enhances resistance to blocking and fingerprinting attacks due to its heterogeneity. TLS incorporates bidirectional nonces in the handshake process, rendering the proxy immune to replay attacks [79]. Moreover, we disabled the older SSL and TLS versions, allowing TLS 1.2 and higher to take advantage of more secure data transmission and stronger encryption. Furthermore, we disabled 0-RTT (zero round-trip time). This feature establishes a connection with no round trip zero round-trip time in TLS 1.3, ideally allowing our proxy to match other systems' speeds [84]. However, using 0-RTT can compromise security through reduced forward secrecy, which protects past sessions against future compromises of secret keys. 0-RTT can potentially be used by censors to identify proxy use due to its rare use on the Internet and vulnerability to replay attacks [79]. Figure 10 shows the longer TLS connection establishment duration when using our proxy implementation compared to a direct connection.
- **Rate Limiting:** We integrated rate limiting into our proxy implementation, a technique used to control the amount of incoming and outgoing traffic to and from a network or service. We aim to prevent abuse or overuse of resources, ensuring that a service remains available and

[iii] `https://github.com/sarohusaini/ProxyPro`

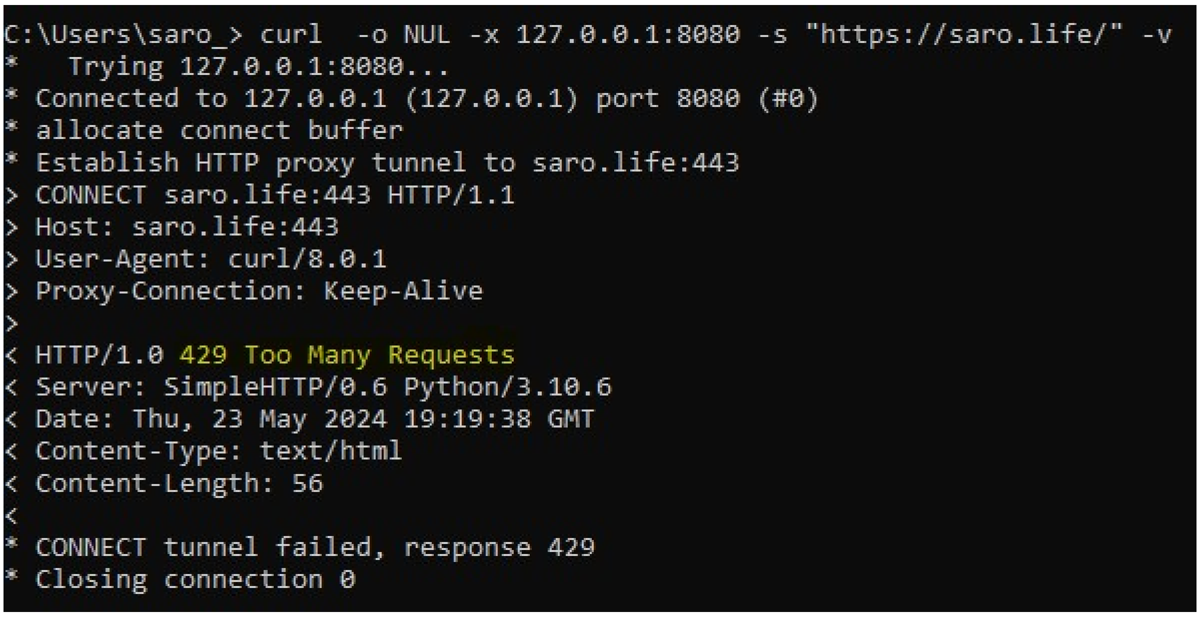


```
C:\Users\saro_> curl -w "@curl-format.txt" -o NUL -s "https://saro.life/"
TLS Handshake Time:  0.296692 seconds
Pretransfer Time:    0.296918 seconds
---------------------------------------------
Time To First Byte: 0.438451s

C:\Users\saro_> curl -w "@curl-format.txt" -o NUL -x 127.0.0.1:8080 -s "https://saro.life/"
TLS Handshake Time:  0.070519 seconds
Pretransfer Time:    0.070673 seconds
---------------------------------------------
Time To First Byte: 0.564972s
```

Figure 10: Proxy and Direction connection TTFB speed.

responsive. Rate limiting is one solution for the denial-of-service and resource exhaustion attacks discussed in the security analysis chapter. By limiting the number of requests, rate limiting helps mitigate DoS attacks if an attacker tries to overwhelm the system with a high volume of requests. We ensure that no single user or IP address can monopolize the server resources. When the number of requests exceeds the defined threshold, additional requests are either delayed, dropped, or rejected, often with an appropriate error message (e.g., HTTP 429 Too Many Requests). In Figure 11, we observe rate limiting in action as we probed the system with many requests. Initially, access to the webpage was available as usual, but once the rate limit was exceeded, access was denied, and the system responded with an error message.



```
C:\Users\saro_> curl  -o NUL -x 127.0.0.1:8080 -s "https://saro.life/" -v
*   Trying 127.0.0.1:8080...
* Connected to 127.0.0.1 (127.0.0.1) port 8080 (#0)
* allocate connect buffer
* Establish HTTP proxy tunnel to saro.life:443
> CONNECT saro.life:443 HTTP/1.1
> Host: saro.life:443
> User-Agent: curl/8.0.1
> Proxy-Connection: Keep-Alive
>
< HTTP/1.0 429 Too Many Requests
< Server: SimpleHTTP/0.6 Python/3.10.6
< Date: Thu, 23 May 2024 19:19:38 GMT
< Content-Type: text/html
< Content-Length: 56
<
* CONNECT tunnel failed, response 429
* Closing connection 0
```

Figure 11: Rate Limiting.

- **IP Blocking:** Another integrated security measure is IP blocking. We use it to prevent network access from specific IP addresses that have been identified as malicious or suspicious. The system can protect itself from potential threats and misuse by blocking these IP addresses. The system compiles a list of IP addresses known for malicious activities based on security reports, threat intelligence, or abnormal behavior patterns. It continuously monitors incoming traffic to detect requests from these IP addresses. It automatically denies requests from identified IP addresses, often responding with an appropriate error message (e.g., HTTP 403 Forbidden).

- **Plausible Response to Probes:** When the proxy server is probed by a censor, it must respond in a way that does not reveal its proxy functionality. We integrated common error pages, such as *404 Not Found and 403 Forbidden*, which are prevalent online and provide legitimate-looking responses to probes. These error pages mimic standard server behavior, making it difficult for censors to distinguish the proxy server from a regular web server. Figure 12 depicts this probe mitigation mechanism.

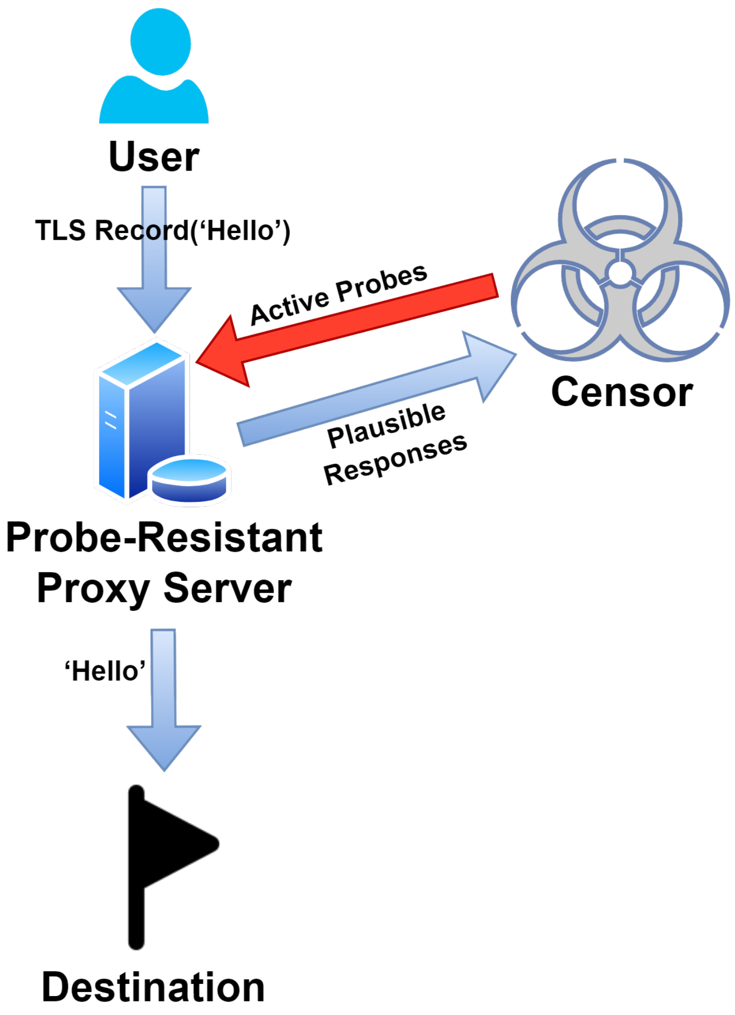


Figure 12: Active Probe mitigation from the Probe-Resistant Proxy

7.2 Computational Costs and Overhead

Using a proxy or similar intermediary tool can impact the speed of connection establishment. Nonetheless, the minor reduction in speed is often worth the added security and privacy benefits. Since we currently do not have a complete implementation of ProxyPro to test the additional overhead it adds compared to a direct connection, it was essential to conduct tests and use secondary available data to obtain an estimated overhead. We evaluated the performance of various components and operations necessary for establishing a direct connection. Additionally, we utilized the best available data on the overhead introduced by a proxy shuffling algorithm and used data from tests we conducted on our partial implementation of the probe-resistant proxy. By measuring the durations of processes involved in establishing a non-proxied connection and those of the additional components integrated by us, we can estimate a potential overhead our scheme might add to regular connection.

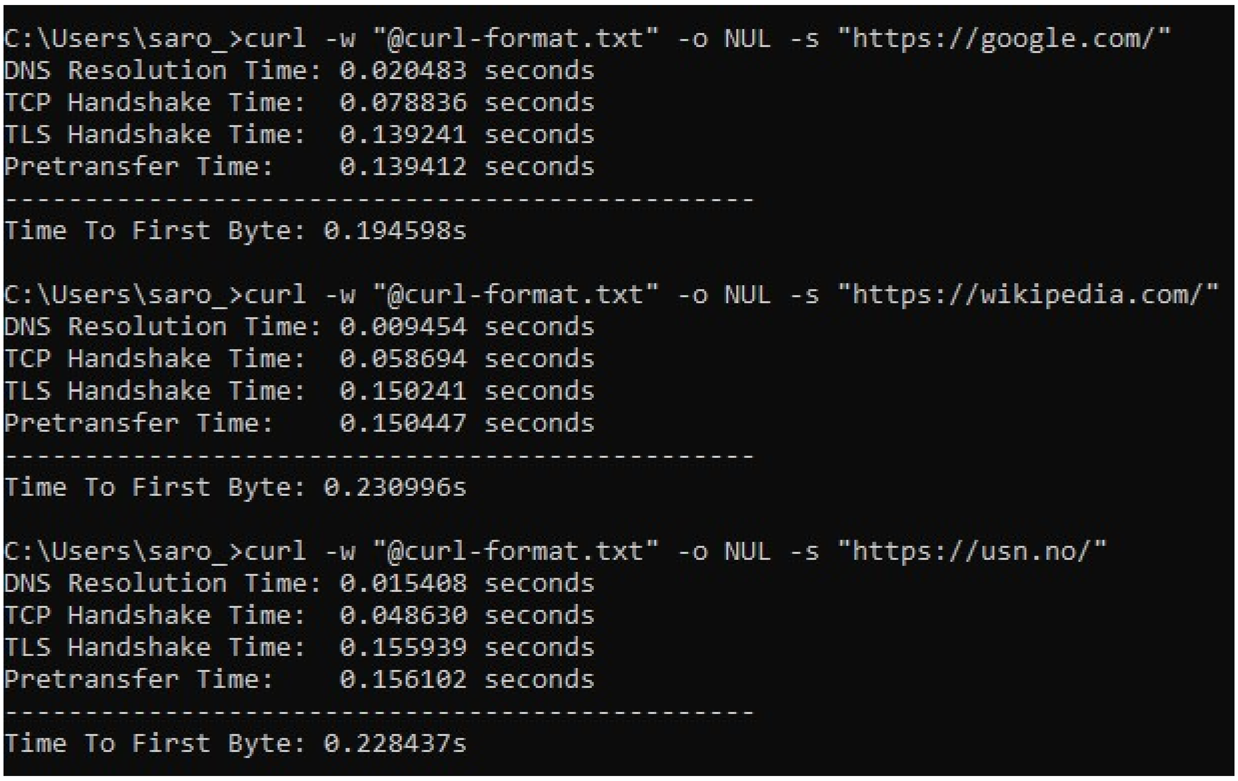
Several factors can impact the range of performance metrics and overhead associated with establishing either a regular Internet connection or a connection through a proxy, including countries where the tests were conducted, the Internet Service Providers, Content Delivery Networks, the resolver providers, the protocols and encryption methods involved, etc [51]. Below, we detail the performance numbers of different workflows and steps involved in both direct connection establishment and ProxyPro. We explore and clarify these processes and provide an overall estimated overhead added by ProxyPro.

The processes involved in a regular connection establishment are relatively straight forward. The processes for getting to the Time to First byte (TTFB) typically include the essential DNS lookup, TCP connection establishment, TLS handshake if the connection is secured, and Pretransfer time [51, 62].

- **DNS Resolution:** The DNS lookup is an essential connection establishment process. In this process, the DNS resolver queries the authoritative name server to obtain the IP address associated with a domain name. The performance impact varies significantly based on geographic location, network infrastructure, and resolver choice [52]. We tested the DNS resolution times of three different domains, as shown in Figure 13. The query times are 9.5 ms, 15.5 ms, and 20 ms respectively. We understand that the processing times of different DNS types across multiple domains are not far from each other. Using a median value between 10-20 ms is an accurate estimated DNS resolution value for our calculations.
- **TCP Connection Establishment:** Performing the 3-way TCP handshake is another important step of establishing a reliable Internet connection [51]. We measured the time for establishing the TCP session of the same three domains as shown in Figure 13. The numbers

show 48 ms, 58 ms, and 78 ms, respectively.

- **TLS Connection Establishment:** The TLS handshake is a critical part of establishing a secure TLS connection. TLS encrypts the data exchanged between the client and server, preventing eavesdroppers from reading the contents of the communication [7]. Our tests showed the TLS session establishment times of 139 ms, 150 ms, and 156 ms, respectively. Showing a small difference across the three domains.
- **Pretransfer Time:** There is a short processing time called the Pretransfer time. This is the time just up to the TTFB. Our test showed a similar Pretransfer median of around 150 ms, as shown in Figure 13.



```
C:\Users\saro_>curl -w "@curl-format.txt" -o NUL -s "https://google.com/"
DNS Resolution Time: 0.020483 seconds
TCP Handshake Time:  0.078836 seconds
TLS Handshake Time:  0.139241 seconds
Pretransfer Time:    0.139412 seconds
-----------------------------------------------
Time To First Byte: 0.194598s

C:\Users\saro_>curl -w "@curl-format.txt" -o NUL -s "https://wikipedia.com/"
DNS Resolution Time: 0.009454 seconds
TCP Handshake Time:  0.058694 seconds
TLS Handshake Time:  0.150241 seconds
Pretransfer Time:    0.150447 seconds
-----------------------------------------------
Time To First Byte: 0.230996s

C:\Users\saro_>curl -w "@curl-format.txt" -o NUL -s "https://usn.no/"
DNS Resolution Time: 0.015408 seconds
TCP Handshake Time:  0.048630 seconds
TLS Handshake Time:  0.155939 seconds
Pretransfer Time:    0.156102 seconds
-----------------------------------------------
Time To First Byte: 0.228437s
```

Figure 13: Direct connection tests on our personal device.

Table 9: Estimated time for a direct connection establishment.

Connection Workflow	**Direct Connection**
DNS Resolution	~10-15 ms
TCP Connection Establishment	~48-78 ms
TLS Connection Establishment	~139-156 ms
Estimated Time to First Byte (TTFB)	**~194-228 ms**

The ProxyPro scheme introduces additional processes to a standard connection establishment, inevitably resulting in some unavoidable but manageable latency. The shuffling algorithm is one of the scheme's most eminent features, adding overhead to the processing time. Our probe-resistant proxy service is another component that may introduce some latency.

- **Shuffle Mapping:** The speed of a proxy shuffling process is primarily determined by the time it takes to perform mapping updates to the IP and DNS mapping tables. Larger network sizes can increase the time of the shuffling process. However, tests on the largest test have shown negligible speed difference [62]. The time grows roughly linearly with the number of subdomains and online services. The most significant shuffling performance can be achieved by hardware and software optimization. Kon et al. [62] managed to reduce the overall shuffling time to just 1100 ms from 6900 ms by utilizing a local Dell PowerEdge R350 machine instead of an outsourced cloud-based DNS service. Moreover, replacing their default Python bindings with a C++ API also contributed to this improvement. We believe that we can achieve an overall shuffle speed of approximately 1000 ms with appropriate software and hardware optimizations in the future.
- **Proxy Establishment:** The probe-resistant proxy service can add extra overhead to the connection establishment time. As outlined in the design chapter, this proxy is installed on a proxy server situated between the shuffling switch and the web server responsible for authenticating the user and forwarding their connection. Our tests on our probe-resistant proxy implementation showed an average of 120-130 ms on top of a direct connection establishment, as shown in Figure 10.

In summary, our tests on direct connections show an average of approximately 194-228 ms TTFB as presented in Table 9, which is the standard procedure for calculating connection establishment times [79]. ProxyPro introduces additional components and processes to the normal connection establishment procedure, resulting in an estimated overall overhead of around 1152 ms, as shown in Table 10. This overhead is acceptable given the numerous advantages provided by ProxyPro, especially considering that connection establishment delays can vary widely from under 100 ms to 62,071 ms due to various factors [85].

Table 10: Estimated overhead added by ProxyPro.

Connection Workflow	**ProxyPro**
Shuffle Mapping	~1100 ms
Proxy Establishment	~150-170
Total Estimated Overhead	**~1152 ms**

8 Conclusion & Future Work

Governments and other entities become more sophisticated in their approaches to Internet censorship, employing sophisticated technologies to filter and block content. The need for equally advanced circumvention methods has become more pressing. This technological arms race makes it imperative for researchers, activists, and developers to continuously innovate and share knowledge to ensure that the Internet remains open and accessible.

We have systematically explored the landscape of Internet censorship and the various techniques developed to circumvent such restrictions. Through the investigation of current censorship mechanisms employed by various regimes worldwide and an in-depth analysis of circumvention technologies, this thesis has shed light on the ongoing battle for unrestricted Internet access. Our research underscores the dynamic and evolving nature of both Internet censorship and the efforts to bypass these restrictions, highlighting the innovative approaches developed to stay one step ahead of censors.

This thesis's significant contributions include the introduction of ProxyPro, a circumvention scheme that integrates multiple state-of-the-art techniques to provide a robust solution and contribute to the struggle against Internet censorship. We developed a probe-resistant proxy, which is an important component of the complete ProxyPro design. This proxy service can be deployed independently and function as a standalone solution. Furthermore, we have identified critical gaps and answered the posed research questions. The detailed examination of both censorship techniques and circumvention tools has provided valuable insights into the strengths and weaknesses of current methods, informing the development of more effective strategies for ensuring open and free access to the Internet.

8.1 Future Work

We aim to continue improving our design to align with future advancements and fully realize the system's development. The following points specify our coding development and real-world deployment aims for implementing ProxyPro.

- **Real-world Deployment:** It is understandable that developing and wide deployment of a censorship circumvention system is not easy and is mostly dominated by worldwide volunteered projects and big companies with adequate resources (e.g., Tor, VPNs, etc). The proposed ProxyPro design and similar methods, backed by a limited number of university researchers, are mostly proposed techniques or prototypes deployed in university and private testbeds. It would be highly valuable if ProxyPro and other censorship circumvention techniques were deployed in public to test their real-world capabilities. Prioritizing the deployment of ProxyPro for public use is crucial to fulfill its primary purpose of helping censored users and contributing to the fight against Internet censorship.

- **Volunteer Testers:** Recruiting volunteers in censored regions to test the real-world circumvention to evaluate the effectiveness of ProxyPro and other anti-censorship tools in real-world scenarios is crucial. Testing the effectiveness of ProxyPro in the wild and by volunteers in censored countries should be a major future priority. However, given censorship's sensitive and political nature, ethical considerations in conducting such research are essential. Primarily, it is important to ensure the safety of volunteers participating in censorship measurement efforts, particularly in countries with strict Internet regulations where they may face legal repercussions.
- **Proxy service deployment:** We aim to enhance our probe-resistant proxy service by adding additional safety and security features, deploying it, and making it available for public use as a standalone project. Additionally, a website could be developed for product downloads and facilitate user donations. These donations would be used for the development and further improvement of ProxyPro.
- **Graphical User Interface:** Users can manually install and configure the proxy; however, including an interactive GUI is essential to assist less tech-savvy users. Our proxy service should incorporate an easy-to-use interface to enhance the quality of service (QoS). This feature can be integrated into the complete proposed scheme for the long term.
- **Code assimilation:** It is important to have the codes for the integrated techniques in a similar programming language or most widely used to help with compatibility and ease of development and implementation. NetShuffle was developed using Python, and the Lox distribution system was implemented in Rust. Assimilation of the codes in one programming language for ease of integration can be one of our steps in the future.

Moreover, various resources, tools, and equipment are needed to implement ProxyPro. Below, we detail the requirements of a project of this nature:

- **A University network and testbed:** Edge networks discussed in Section 3.9, which include environments like university campuses, corporate networks, and private data centers, occupy a unique position within the Internet's overall architecture. They are considered *edge* because they sit at the boundary between the user-facing parts of the Internet and the core networks that form the backbone of the Internet infrastructure. These networks are typically well-equipped with significant resources, including blocks of publicly routed IP address blocks and substantial hardware capabilities such as compute infrastructure and high-speed networking. To implement ProxyPro, it is necessary to have the support of a university's campus network (e.g., USN) to host the prototype and various components of the project.

- **Programmable Switches:** It is required to acquire programmable devices (e.g., Intel Tofino P4 switch) to host the mapping required to achieve the proxy shuffling process.
- **Server Devices:** A regular computer can be configured and act as the server for a network, but in the case of a circumvention system that involves many components and needs to be up and running 24/7 year around, capable computers and dedicated server devices are an essential requirement.

8.1.1 Future Research Suggestions

Looking forward, the fight against Internet censorship is far from over. As censorship methods grow more sophisticated and incorporate advanced technologies like artificial intelligence and machine learning, strategies for circumvention must adapt and evolve accordingly. We highlight several key areas that are important and require further research and future investigation. We mainly focus on a profound study of emerging technologies such as AI and blockchain.

Blockchain technology holds significant promise for creating decentralized communication networks inherently resistant to censorship and surveillance. The works investigated in Section 3.3 represent considerable steps in the practical application of blockchain technology for purposes beyond traditional financial transactions, highlighting its potential in ensuring free and unrestricted communication across the Internet. The ability of blockchain to maintain an immutable ledger without central oversight makes it well-suited for supporting communication channels that stay open and accessible. Given these features and the technology's novelty, additional research is very valuable for developing more robust, clear, and decentralized methods of exchanging information.

Furthermore, reports show that governments are leveraging generative AI technologies to tighten online censorship and amplify surveillance, thus making these controls more efficient and effective. It was recently identified that countries use generative AI to manipulate information on political and social issues, and automated systems for content moderation, enforced by censorship laws, are required in at least 22 countries [86]. In response, advancements in artificial intelligence should be leveraged to automate the discovery of circumvention pathways and optimize the performance of existing tools. The potential of AI can serve beneficial purposes such as proper regulation, strong data privacy laws, advancements in detecting misinformation, and the role of bypassing government censorship.

Another promising area of future research involves exploring the potential of satellite Internet systems to provide uncensored Internet access in regions where traditional methods of communication are heavily monitored and restricted. Activists have already started employing publicly available satellite Internet constellations such as Starlink to bypass censorship, and oppressive regimes have already started taking action against such companies [87]. Deploying low Earth orbit satellite constellations for cheap and private use could revolutionize Internet access, offering a reliable alternative that circumvents terrestrial censorship infrastructure. It is important that researchers and activists further explore this relatively untapped field and collaborate with satellite Internet providers who are willing to contribute to Internet censorship circumvention efforts.

The fight for uncensored Internet access is an ongoing struggle that requires continuous innovation and collaboration. By leveraging emerging technologies and building on the foundations laid by current circumvention techniques, future research can contribute to developing more effective strategies for overcoming Internet censorship, ultimately ensuring that the right to free and open access to information is upheld worldwide.

References

[1] P. H. Ang and B. Nadarjan, "Censorship and the internet: A singapore perspective," *Commun. ACM*, vol. 39, no. 6, pp. 72–78, 1996.

[2] E. Goldstein, "The internet in the mideast and north africa: Free expression and censorship," *Human Rights Watch*, 1999.

[3] A. L. Shapiro, "The internet," *Foreign Policy*, no. 115, pp. 14–27, 1999.

[4] J. L. Zittrain, R. Faris, H. Noman, J. Clark, C. Tilton, and R. Morrison-Westphal, "The shifting landscape of global internet censorship," *Berkman Klein Center Research Publication*, no. 2017-4, pp. 17–38, 2017.

[5] J. L. Hall, M. D. Aaron, A. Andersdotter, B. Jones, N. Feamster, and M. Knodel, "A Survey of Worldwide Censorship Techniques," Internet-Draft draft-irtf-pearg-censorship-10, Internet Engineering Task Force, Mar. 2023. Work in Progress.

[6] P. Liubinskii, "The Great Firewall's active probing circumvention technique with port knocking and SDN," master's thesis, Aalto University. School of Electrical Engineering, 2021.

[7] S. Frolov and E. Wustrow, "The use of TLS in censorship circumvention," in *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*, The Internet Society, 2019.

[8] A. Master and C. Garman, "A worldwide view of nation-state internet censorship," *Free and Open Communications on the Internet*, pp. 1–21, 2023.

[9] D. E. E. III, "Transport layer security (TLS) extensions: Extension definitions," *RFC*, vol. 6066, pp. 1–25, 2011.

[10] W. M. Shbair, T. Cholez, A. Goichot, and I. Chrisment, "Efficiently bypassing sni-based HTTPS filtering," in *IFIP/IEEE International Symposium on Integrated Network Management, IM 2015, Ottawa, ON, Canada, 11-15 May, 2015* (R. Badonnel, J. Xiao, S. Ata, F. D. Turck, V. Groza, and C. R. P. dos Santos, eds.), pp. 990–995, IEEE, 2015.

[11] Z. Chai, A. Ghafari, and A. Houmansadr, "On the importance of {Encrypted-SNI}({{{{{ESNI}}}}}) to censorship circumvention," in *9th USENIX Workshop on Free and Open Communications on the Internet (FOCI 19)*, 2019.

[12] S. Satija and R. Chatterjee, “BlindTLS: Circumventing TLS-based HTTPS censorship,” in *Proceedings of the ACM SIGCOMM 2021 Workshop on Free and Open Communications on the Internet*, pp. 43–49, 2021.

[13] E. Rescorla, “The transport layer security (TLS) protocol version 1.3,” *RFC*, vol. 8446, pp. 1–160, 2018.

[14] M. K. Land, “Against privatized censorship: Proposals for responsible delegation,” *Va. J. Int’l L.*, vol. 60, p. 363, 2019.

[15] E. Rescorla, “Overview of apple’s client-side csam scanning,” *Educated Guesswork*, 2021.

[16] R. Bendrath, “Global technology trends and national regulation: Explaining variation in the governance of deep packet inspection,” in *Global technology trends and national regulation: Explaining Variation in the Governance of Deep Packet Inspection*, 2009.

[17] J. C. Park and J. R. Crandall, “Empirical study of a national-scale distributed intrusion detection system: Backbone-level filtering of HTML responses in china,” in *2010 International Conference on Distributed Computing Systems, ICDCS 2010, Genova, Italy, June 21-25, 2010*, pp. 315–326, IEEE Computer Society, 2010.

[18] Z. Weinberg, D. Barradas, and N. Christin, “Chinese wall or swiss cheese? keyword filtering in the great firewall of china,” in *WWW ’21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021* (J. Leskovec, M. Grobelnik, M. Najork, J. Tang, and L. Zia, eds.), pp. 472–483, ACM / IW3C2, 2021.

[19] K. Elmenhorst, B. Schütz, N. Aschenbruck, and S. Basso, “Web censorship measurements of HTTP/3 over QUIC,” in *IMC ’21: ACM Internet Measurement Conference, Virtual Event, USA, November 2-4, 2021* (D. Levin, A. Mislove, J. Amann, and M. Luckie, eds.), pp. 276–282, ACM, 2021.

[20] E. Hjelmvik and W. John, “Breaking and Improving Protocol Obfuscation,” *Chalmers University of Technology, Tech. Rep*, vol. 123751, 2010.

[21] S. Aryan, H. Aryan, and J. A. Halderman, “Internet censorship in iran: A first look,” in *3rd USENIX Workshop on Free and Open Communications on the Internet, FOCI ’13, Washington, D.C., USA, August 13, 2013* (J. R. Crandall and J. Wright, eds.), USENIX Association, 2013.

[22] P. Winter and S. Lindskog, “How china is blocking tor,” *CoRR*, vol. abs/1204.0447, 2012.

[23] P. E. Hoffman and P. McManus, “DNS queries over HTTPS (doh),” *RFC*, vol. 8484, pp. 1–21, 2018.

[24] Z. Hu, L. Zhu, J. S. Heidemann, A. Mankin, D. Wessels, and P. E. Hoffman, “Specification for DNS over transport layer security (TLS),” *RFC*, vol. 7858, pp. 1–19, 2016.

[25] L. Jin, S. Hao, H. Wang, and C. Cotton, “Understanding the impact of encrypted DNS on internet censorship,” in *WWW ’21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021* (J. Leskovec, M. Grobelnik, M. Najork, J. Tang, and L. Zia, eds.), pp. 484–495, ACM / IW3C2, 2021.

[26] P. Zhu, K. Man, Z. Wang, Z. Qian, R. Ensafi, J. A. Halderman, and H. Duan, “Characterizing transnational internet performance and the great bottleneck of china,” *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 4, no. 1, pp. 13:1–13:23, 2020.

[27] S. Cho, R. Fontugne, K. Cho, A. Dainotti, and P. Gill, “BGP hijacking classification,” in *Network Traffic Measurement and Analysis Conference, TMA 2019, Paris, France, June 19-21, 2019* (S. Secci, I. Chrisment, M. Fiore, L. Tabourier, and K. Lim, eds.), pp. 25–32, IEEE, 2019.

[28] R. R. Brooks, L. Yu, I. Özçelik, J. Oakley, and N. Tusing, “Distributed denial of service (ddos): A history,” *IEEE Ann. Hist. Comput.*, vol. 44, no. 2, pp. 44–54, 2022.

[29] R. Clayton, S. J. Murdoch, and R. N. M. Watson, “Ignoring the great firewall of china,” in *Privacy Enhancing Technologies, 6th International Workshop, PET 2006, Cambridge, UK, June 28-30, 2006, Revised Selected Papers* (G. Danezis and P. Golle, eds.), vol. 4258 of *Lecture Notes in Computer Science*, pp. 20–35, Springer, 2006.

[30] Anonymous, “Towards a comprehensive picture of the great firewall’s DNS censorship,” in *4th USENIX Workshop on Free and Open Communications on the Internet, FOCI ’14, San Diego, CA, USA, August 18, 2014* (J. R. Crandall and V. Paxson, eds.), USENIX Association, 2014.

[31] L. Ruan, J. Knockel, and M. Crete-Nishihata, “Censored contagion: How information on the coronavirus is managed on chinese social media,” tech. rep., The Citizen Lab, University of Toronto, 2020.

[32] K. Singh, G. Grover, and V. Bansal, “How india censors the web,” *CoRR*, vol. abs/1912.08590, 2019.

[33] P. Winter, *Measuring and circumventing Internet censorship*. PhD thesis, Karlstads universitet, 2014.

[34] Z. Tsiatsikas, G. Karopoulos, and G. Kambourakis, “Measuring the adoption of TLS encrypted client hello extension and its forebear in the wild,” in *Computer Security. ESORICS 2022 International Workshops - CyberICPS 2022, SECPRE 2022, SPOSE 2022, CPS4CIP 2022,*

CDT&SECOMANE 2022, EIS 2022, and SecAssure 2022, Copenhagen, Denmark, September 26-30, 2022, Revised Selected Papers (S. K. Katsikas, F. Cuppens, C. Kalloniatis, J. Mylopoulos, F. Pallas, J. Pohle, M. A. Sasse, H. Abie, S. Ranise, L. Verderame, E. Cambiaso, J. M. Vidal, M. A. S. Monge, M. Albanese, B. Katt, S. Pirbhulal, and A. Shukla, eds.), vol. 13785 of *Lecture Notes in Computer Science*, pp. 177–190, Springer, 2022.

[35] J. Verkamp and M. Gupta, "Inferring mechanics of web censorship around the world," in *2nd USENIX Workshop on Free and Open Communications on the Internet, FOCI '12, Bellevue, WA, USA, August 6, 2012* (R. Dingledine and J. Wright, eds.), USENIX Association, 2012.

[36] Z. Nabi, "The anatomy of web censorship in pakistan," in *3rd USENIX Workshop on Free and Open Communications on the Internet, FOCI '13, Washington, D.C., USA, August 13, 2013* (J. R. Crandall and J. Wright, eds.), USENIX Association, 2013.

[37] F. House, "Freedom on the net 2021: The global drive to control big tech," *Freedom House: Washington, DC, USA*, 2021.

[38] K. Bock, Y. Fax, K. Reese, J. Singh, and D. Levin, "Detecting and evading censorship-in-depth: A case study of iran's protocol whitelister," in *10th USENIX Workshop on Free and Open Communications on the Internet, FOCI 2020, August 11, 2020* (R. Ensafi and H. Klein, eds.), USENIX Association, 2020.

[39] K. Elmenhorst, "A Quick Look at QUIC Censorship," *Open Technology Fund*, 2022.

[40] V. Ververis, *Internet censorship in the European Union*. PhD thesis, Humboldt-Universität zu Berlin, Wirtschaftswissenschaftliche Fakultät, 2023.

[41] D. Anderson, "Splinternet behind the great firewall of china: Once china opened its door to the world, it could not close it again.," *Queue*, vol. 10, p. 40–49, nov 2012.

[42] P. Pearce, B. Jones, F. Li, R. Ensafi, N. Feamster, N. Weaver, and V. Paxson, "Global measurement of DNS manipulation," in *26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017* (E. Kirda and T. Ristenpart, eds.), pp. 307–323, USENIX Association, 2017.

[43] N. Feamster, M. Balazinska, G. Harfst, H. Balakrishnan, and D. R. Karger, "Infranet: Circumventing web censorship and surveillance," in *Proceedings of the 11th USENIX Security Symposium, San Francisco, CA, USA, August 5-9, 2002* (D. Boneh, ed.), pp. 247–262, USENIX, 2002.

[44] D. Fifield, N. Hardison, J. D. Ellithorpe, E. Stark, D. Boneh, R. Dingledine, and P. A. Porras, "Evading censorship with browser-based proxies," in *Privacy Enhancing Technologies - 12th*

International Symposium, PETS 2012, Vigo, Spain, July 11-13, 2012. Proceedings (S. Fischer-Hübner and M. K. Wright, eds.), vol. 7384 of *Lecture Notes in Computer Science*, pp. 239–258, Springer, 2012.

[45] P. K. Sharma, D. Gosain, and S. Chakravarty, "Camoufler: Accessing the censored web by utilizing instant messaging channels," in *ASIA CCS '21: ACM Asia Conference on Computer and Communications Security, Virtual Event, Hong Kong, June 7-11, 2021* (J. Cao, M. H. Au, Z. Lin, and M. Yung, eds.), pp. 147–161, ACM, 2021.

[46] D. Ding, K. Jeong, S. Xing, M. Conti, R. Figueiredo, and F. Liu, "Send: A social network friendship enhanced decentralized system to circumvent censorships," *IEEE Transactions on Services Computing*, vol. 15, no. 1, pp. 346–360, 2022.

[47] M. Nasr, H. Zolfaghari, A. Houmansadr, and A. Ghafari, "Massbrowser: Unblocking the censored web for the masses, by the masses," in *27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020*, The Internet Society, 2020.

[48] D. Fifield, *Threat modeling and circumvention of Internet censorship*. PhD thesis, University of California, Berkeley, USA, 2017.

[49] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. R. Iyengar, J. Bailey, J. Dorfman, J. Roskind, J. Kulik, P. Westin, R. Tenneti, R. Shade, R. Hamilton, V. Vasiliev, W. Chang, and Z. Shi, "The QUIC transport protocol: Design and internet-scale deployment," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication, SIGCOMM 2017, Los Angeles, CA, USA, August 21-25, 2017*, pp. 183–196, ACM, 2017.

[50] N. P. Hoang, M. Polychronakis, and P. Gill, "Measuring the accessibility of domain name encryption and its impact on internet filtering," in *Passive and Active Measurement - 23rd International Conference, PAM 2022, Virtual Event, March 28-30, 2022, Proceedings* (O. Hohlfeld, G. C. M. Moura, and C. Pelsser, eds.), vol. 13210 of *Lecture Notes in Computer Science*, pp. 518–536, Springer, 2022.

[51] A. Hounsel, P. Schmitt, K. Borgolte, and N. Feamster, "Can encrypted DNS be fast?," in *Passive and Active Measurement - 22nd International Conference, PAM 2021, Virtual Event, March 29 - April 1, 2021, Proceedings* (O. Hohlfeld, A. Lutu, and D. Levin, eds.), vol. 12671 of *Lecture Notes in Computer Science*, pp. 444–459, Springer, 2021.

[52] R. Chhabra, P. Murley, D. Kumar, M. D. Bailey, and G. Wang, "Measuring dns-over-https performance around the world," in *IMC '21: ACM Internet Measurement Conference, Virtual Event, USA, November 2-4, 2021* (D. Levin, A. Mislove, J. Amann, and M. Luckie, eds.), pp. 351–365, ACM, 2021.

[53] G. Shi, "Multiproxy: a collaborative approach to censorship circumvention," *Delft University of Technology*, 2019.

[54] N. Alsalami and B. Zhang, "Utilizing public blockchains for censorship-circumvention and iot communication," in *2019 IEEE Conference on Dependable and Secure Computing, DSC 2019, Hangzhou, China, November 18-20, 2019*, pp. 1–7, IEEE, 2019.

[55] K. Bock, G. Hughey, X. Qiang, and D. Levin, "Geneva: Evolving censorship evasion strategies," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019* (L. Cavallaro, J. Kinder, X. Wang, and J. Katz, eds.), pp. 2199–2214, ACM, 2019.

[56] K. Bock, G. Hughey, L. Merino, T. Arya, D. Liscinsky, R. Pogosian, and D. Levin, "Come as you are: Helping unmodified clients bypass censorship with server-side evasion," in *SIGCOMM '20: Proceedings of the 2020 Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication, Virtual Event, USA, August 10-14, 2020* (H. Schulzrinne and V. Misra, eds.), pp. 586–598, ACM, 2020.

[57] M. Wu, J. Sippe, D. Sivakumar, J. Burg, P. Anderson, X. Wang, K. Bock, A. Houmansadr, D. Levin, and E. Wustrow, "How the great firewall of china detects and blocks fully encrypted traffic," in *32nd USENIX Security Symposium, USENIX Security 2023, Anaheim, CA, USA, August 9-11, 2023* (J. A. Calandrino and C. Troncoso, eds.), pp. 2653–2670, USENIX Association, 2023.

[58] D. Fifield, "Turbo tunnel, a good way to design censorship circumvention protocols," in *10th USENIX Workshop on Free and Open Communications on the Internet, FOCI 2020, August 11, 2020* (R. Ensafi and H. Klein, eds.), USENIX Association, 2020.

[59] J. A. Donenfeld, "Wireguard: Next generation kernel network tunnel," in *24th Annual Network and Distributed System Security Symposium, NDSS 2017, San Diego, California, USA, February 26 - March 1, 2017*, The Internet Society, 2017.

[60] D. Barradas, N. Santos, L. E. T. Rodrigues, and V. Nunes, "Poking a hole in the wall: Efficient censorship-resistant internet communications by parasitizing on webrtc," in *CCS '20: 2020*

ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020 (J. Ligatti, X. Ou, J. Katz, and G. Vigna, eds.), pp. 35–48, ACM, 2020.

[61] G. Figueira, D. Barradas, and N. Santos, "Stegozoa: Enhancing webrtc covert channels with video steganography for internet censorship circumvention," in *ASIA CCS '22: ACM Asia Conference on Computer and Communications Security, Nagasaki, Japan, 30 May 2022 - 3 June 2022* (Y. Suga, K. Sakurai, X. Ding, and K. Sako, eds.), pp. 1154–1167, ACM, 2022.

[62] P. Kon, A. Gattani, D. Saharia, T. Cao, D. Barradas, A. Chen, M. Sherr, and B. E. Ujcich, "Netshuffle: Circumventing censorship with shuffle proxies at the edge," in *2024 IEEE Symposium on Security and Privacy (SP),* (Los Alamitos, CA, USA), pp. 36–36, IEEE Computer Society, may 2024.

[63] F. Platzer, M. Schäfer, and M. Steinebach, "Critical traffic analysis on the tor network," in *ARES 2020: The 15th International Conference on Availability, Reliability and Security, Virtual Event, Ireland, August 25-28, 2020* (M. Volkamer and C. Wressnegger, eds.), pp. 77:1–77:10, ACM, 2020.

[64] A. Dunna, C. O'Brien, and P. Gill, "Analyzing china's blocking of unpublished tor bridges," in *8th USENIX Workshop on Free and Open Communications on the Internet, FOCI 2018, Baltimore, MD, USA, August 14, 2018* (L. Gill and R. Jansen, eds.), USENIX Association, 2018.

[65] J. Karlin, D. Ellard, A. W. Jackson, C. E. Jones, G. Lauer, D. Mankins, and W. T. Strayer, "Decoy routing: Toward unblockable internet communication," in *USENIX Workshop on Free and Open Communications on the Internet, FOCI '11, San Francisco, CA, USA, August 8, 2011* (N. Feamster and W. Lee, eds.), USENIX Association, 2011.

[66] D. Fifield, C. Lan, R. Hynes, P. Wegmann, and V. Paxson, "Blocking-resistant communication through domain fronting," *Proc. Priv. Enhancing Technol.*, vol. 2015, no. 2, pp. 46–64, 2015.

[67] C. Bocovich and I. Goldberg, "Secure asymmetry and deployability for decoy routing systems," *Proc. Priv. Enhancing Technol.*, vol. 2018, no. 3, pp. 43–62, 2018.

[68] M. Wei, "Domain shadowing: Leveraging content delivery networks for robust blocking-resistant communications," in *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021* (M. D. Bailey and R. Greenstadt, eds.), pp. 3327–3343, USENIX Association, 2021.

[69] K. Subramani, R. Perdisci, and P. Skafidas, "Measuring cdns susceptible to domain fronting," *CoRR*, vol. abs/2310.17851, 2023.

[70] Y. Govil, L. Wang, and J. Rexford, "MIMIQ: masking ips with migration in QUIC," in *10th USENIX Workshop on Free and Open Communications on the Internet, FOCI 2020, August 11, 2020* (R. Ensafi and H. Klein, eds.), USENIX Association, 2020.

[71] X. Shi, Y. Guo, Y. Wang, and W. Bai, "Mds coding enabled proxy-based internet censorship circumvention system," *Electronics Letters*, vol. 59, no. 13, p. e12858, 2023.

[72] M. AlSabah and I. Goldberg, "Performance and security improvements for tor: A survey," *ACM Comput. Surv.*, vol. 49, no. 2, pp. 32:1–32:36, 2016.

[73] Z. Gao, F. Chen, Y. Wang, W. He, X. Shi, and G. Xie, "Mvpn: A defense architecture against vpn traffic hijacking based on mtd," *Electronics*, vol. 12, no. 3, 2023.

[74] J. A. Vilalonga, J. S. Resende, and H. Domingos, "Torkameleon: Improving tor's censorship resistance with k-anonimization and media-based covert channels," *CoRR*, vol. abs/2303.17544, 2023.

[75] L. Tulloch and I. Goldberg, "Lox: Protecting the social graph in bridge distribution," *Proc. Priv. Enhancing Technol.*, vol. 2023, no. 1, pp. 494–509, 2023.

[76] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "Openflow random host mutation: transparent moving target defense using software defined networking," in *Proceedings of the first workshop on Hot topics in software defined networks, HotSDN@SIGCOMM 2012, Helsinki, Finland, August 13, 2012* (N. Feamster and J. Rexford, eds.), pp. 127–132, ACM, 2012.

[77] T. E. Carroll, M. B. Crouse, E. W. Fulp, and K. S. Berenhaut, "Analysis of network address shuffling as a moving target defense," in *IEEE International Conference on Communications, ICC 2014, Sydney, Australia, June 10-14, 2014*, pp. 701–706, IEEE, 2014.

[78] Y. Xie, G. Gou, G. Xiong, Z. Li, and M. Cui, "Covertness analysis of snowflake proxy request," in *26th International Conference on Computer Supported Cooperative Work in Design, CSCWD 2023, Rio de Janeiro, Brazil, May 24-26, 2023* (W. Shen, J. A. Barthès, J. Luo, A. S. Vivacqua, D. Schneider, C. Xie, J. Zhang, H. Zhu, K. Peng, and C. L. R. da Motta, eds.), pp. 1802–1807, IEEE, 2023.

[79] S. Frolov and E. Wustrow, "HTTPT: A probe-resistant proxy," in *10th USENIX Workshop on Free and Open Communications on the Internet, FOCI 2020, August 11, 2020* (R. Ensafi and H. Klein, eds.), USENIX Association, 2020.

[80] I. A. Lovecruft and H. de Valence, "Hyphae: Social secret sharing," 2017.

[81] Q. Wang, Z. Lin, N. Borisov, and N. Hopper, “rbridge: User reputation based tor bridge distribution with privacy preservation,” in *20th Annual Network and Distributed System Security Symposium, NDSS 2013, San Diego, California, USA, February 24-27, 2013*, The Internet Society, 2013.

[82] F. Douglas, Rorshach, W. Pan, and M. Caesar, “Salmon: Robust proxy distribution for censorship circumvention,” *Proc. Priv. Enhancing Technol.*, vol. 2016, no. 4, pp. 4–20, 2016.

[83] Alice, Bob, Carol, J. Beznazwy, and A. Houmansadr, “How china detects and blocks shadowsocks,” in *IMC ’20: ACM Internet Measurement Conference, Virtual Event, USA, October 27-29, 2020*, pp. 111–124, ACM, 2020.

[84] N. Aviram, K. Gellert, and T. Jager, “Session resumption protocols and efficient forward security for TLS 1.3 0-rtt,” *J. Cryptol.*, vol. 34, no. 3, p. 20, 2021.

[85] U. Bauknecht and T. Enderle, “An investigation on core network latency,” in *30th International Telecommunication Networks and Applications Conference, ITNAC 2020, Melbourne, Australia, November 25-27, 2020*, pp. 1–6, IEEE, 2020.

[86] R. Chandran, “Ai ’supercharges’ online disinformation and censorship, report warns,” *Thomson Reuters Foundation*, 2023.

[87] A. Cuthbertson, “Starlink: Iran protesters activate ‘close to 100’ illegal internet receivers, elon musk says,” *The Independent*, 2022.