St. Pölten University of Applied Sciences

/informatik & security

/fh/// st.pölten

Synthetic Content for Probe-Resistant Proxies

Diploma thesis

For attainment of the academic degree of

Diplom-Ingenieur/in

submitted by

Armin Huremagic, BSc MSc

is201819

in the

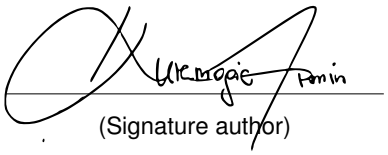
University Course Information Security at St. Pölten University of Applied Sciences

Supervision

Advisor: Dr. Torsten Priebe

Assistance: -

St. Pölten, January 20, 2023



(Signature author) (Signature advisor)

Declaration

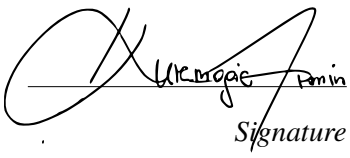
I hereby affirm that

- I have written this thesis independently, that I have not used any sources or aids other than those indicated, and that I have not made use of any unauthorised assistance.
- I have not previously submitted this thesis topic to an assessor, either in Austria or abroad, for evaluation or as an examination paper in any form.
- this thesis corresponds to the thesis assessed by the assessor.

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

20.02.2023

Date



Signature

Kurzfassung

Inspiriert von Technologien wie HTTPT, die Webserver als Proxys nutzen, um die Internetzensur zu umgehen, befasst sich diese Arbeit mit dem Thema Internetzensur und untersucht, wie GPT-3, ein modernes Sprachmodell, verwendet werden kann, um authentisch aussehende Websites zu generieren, die dazu verwendet werden können, resistente Proxys, die für die Umgehung der Internetzensur verwendet werden, als harmlose Webserver zu tarnen. In dieser Arbeit werden zunächst die Beweggründe für die Internetzensur und die verschiedenen Techniken vorgestellt, die von Zensoren verwendet werden, um den Zugang zu Websites zu beschränken, wie z. B. DNS-Manipulation, IP-basierte Sperrungen und wie Protokolle wie auf HTTP- und HTTPS-Verbindungen mit Hilfe von Zensurgeräten eingegriffen werden können. Es werden Methoden zur Umgehung der Zensur erörtert, darunter die Verwendung von Proxys wie Tor-Bridges und Technologien wie Pluggable-Transportprotokolle, mit denen die wahre Absicht des Netzwerkverkehrs verschleiert werden kann. In dieser Arbeit werden Experimente durchgeführt, um zu messen, welche Schlüsselwörter am anfälligsten für Zensur sind, um sie bei der Erstellung von Text- und Bildinhalten mit GPT-3 zu vermeiden. Außerdem werden Experimente mit der GPT-API durchgeführt, um Websites mit Text- und Bildinhalten zu generieren. Insgesamt zielt diese Arbeit darauf ab, das Potenzial von GPT-3 bei der Umgehung der Internetzensur aufzuzeigen und die Herausforderungen, die für einen effektiven Einsatz in diesem Bereich überwunden werden müssen. Die Ergebnisse dieser Forschung legen nahe, dass GPT-3 und andere fortschrittliche KI-Technologien das Potenzial haben, eine wichtige Rolle im Kampf gegen die Internetzensur zu spielen, indem sie denjenigen, die sich für die Meinungsfreiheit und den Zugang zu Informationen im Internet einsetzen, leistungsstarke Werkzeuge zur Verfügung stellen.

Abstract

Inspired by technologies like HTTPT, which leverage web servers as proxies to circumvent internet censorship, this thesis delves into the topic of internet censorship and explores how GPT-3, a state-of-the-art language model, can be used to generate authentic looking websites that can be used to cover probe-resistant proxies used for internet censorship circumvention, harmless web servers. The thesis first introduces the motivations behind internet censorship and the various techniques used by censors to restrict access to websites, such as DNS manipulation, IP based blocking, and how protocols like HTTP and HTTPS can be interfered with using censorship devices. It discusses methods used to circumvent censorship, including the use of proxies like Tor bridges and technologies like pluggable transport protocols that can disguise the true intention of the network traffic. The thesis conducts experiments to measure which keywords are most susceptible to censorship in order to avoid them when generating text and image content using GPT-3. Additionally, experiments with the GPT API are being conducted, to generate websites with text and image content. Overall, this thesis aims to showcase the potential of GPT-3 in helping to circumvent internet censorship and the challenges that must be overcome to effectively use it in this field. The results of this research suggest that GPT-3 and other advanced AI technologies have the potential to play a significant role in the fight against internet censorship, by providing powerful tools for those working to promote freedom of expression and access to information online.

Contents

1 Introduction

This thesis delves into the topic of internet censorship and explores how GPT-3 can be used to generate authentic looking websites that can be used to cover probe-resistant proxies used for internet censorship circumvention, by posing the generated website as a harmless web server. The thesis first introduces the motivations behind internet censorship and the various techniques used by censors to restrict access to certain information, such as DNS manipulation, IP based blocking, and how protocols like HTTP and HTTPS can be interfered with using a censorship device. It discusses methods used to circumvent censorship, including the use of proxies like Tor relays and bridges and technologies like pluggable transport protocols that can disguise the true intention of the network traffic. It introduces GPT-3, a state-of-the-art language model developed by OpenAI, and the technologies behind it such as natural language processing, machine learning, transformer models, and language models. This thesis conducts experiments to measure which keywords are most susceptible to censorship in order to avoid them when generating text and image content using GPT-3. Additionally, experiments with the GPT Python library are being conducted, to generate websites with text and image content using GPT-3. Overall, this thesis aims to showcase the potential of GPT-3 in helping to circumvent internet censorship and the challenges that must be overcome to effectively use it in this field.

1.1 Thesis Outline

This document is organized in several parts. Chapter 1 introduces the topic of internet censorship and the technologies behind GPT-3, the challenges within the censorship landscape and the motivation behind working on this topic. Chapter 2 describes some the fundamental knowledge needed for understanding the internet censorship landscape as well the technologies on which GPT-3 was built on. Chapter 3 lists the related work.

Chapter 4 describes the experiments conducted for this thesis. Chapter 5 concludes the findings.

1.2 Motivation

The motivation behind writing a research paper on the topic of internet censorship and how GPT-3 can be used to generate real-looking websites for censorship circumvention is to address the growing threat of internet censorship and its negative impact on the free flow of information. In today's digital age, the internet has become a crucial tool for the dissemination of knowledge and ideas. However, with the rise of internet censorship, many individuals and groups are being denied access to information and the ability to express themselves freely online. This not only violates basic human rights but also poses a threat to democracy and the free exchange of ideas. Technologies like GPT-3 can play a crucial role in the fight against internet censorship by allowing individuals and organizations to create websites that appear legitimate to censors while providing access to censored information. These tools can help to ensure that the internet remains a free and open space for all, allowing individuals to access the information they need and exercise their right to free speech and expression. The importance of a free and open internet cannot be overstated, and it is crucial that we use all available tools and technologies to protect it.

2 Prerequisites

The following chapter provides a comprehensive overview of the background knowledge on Internet censorship and AI based technologies that are relevant to the current research. The discussion on Internet censorship will cover the various forms of censorship, the reasons behind censorship, and the methods used to bypass censorship. Additionally, the chapter will delve into the use of AI technologies, specifically natural language processing and transformer models. These technologies have been used to create sophisticated generative models of natural language text, like GPT-3, that can be used to provide synthetic content. The chapter will also give an overview of how these technologies have been used in other related studies, thus providing a solid foundation for understanding the current research.

2.1 Internet Censorship

The motivations behind internet censorship vary significantly and can include political repression of dissidents, human rights activists, or individuals who make critical remarks about the state (e.g., China, Iran, Burma/Myanmar), religious control to prevent the spread of ideas deemed heretical or blasphemous (for example in many Arab states), protection of intellectual property rights such as restrictions on illegally downloaded movies and music and cultural limitations that manifest as oppression of ethnic minorities or sexual minorities. Governments often use the excuse of protecting public morality from activities such as pornography or gambling to justify this form of censorship, but more recently combating terrorism has become a popular justification too. Other proponents argue that some degree of censorship is necessary to combat cyber anarchy or reduce crime levels [1]–[3].

When it comes to deciding how much internet censorship should be imposed, governments have a few options at their disposal. Like allowing completely free access to information, prohibiting access altogether (as seen in North Korea), or somewhere between these two extremes - which is usually what they opt for. This conflict between freedom of speech online and territorial laws speaks volumes about how nation-states are struggling with maintaining control over their cyber-territories [1].

However, Villeneuve's research highlights there are sometimes unintentional consequences when implementing strict forms of internet censorship, including reduced innovation rates due to fewer resources being available online, bad publicity leading countries into becoming pariahs on the international stage, lower tourism figures due to negative perceptions etc., all things which policy makers rarely take into consideration before introducing such policies into their country's system [4].

Internet censorship typically involves controlling access levels; functionality; and contents published on the web with varying degrees depending upon its scope and depth. Both difficult tasks given how considering the size of the internet. ISPs employ different methods for attempting this task including content filtering based on keywords, redirection users onto proxy servers and IP address blocking for certain websites being accessed, just to name a few [4]–[7].

The organization Freedom House [8] ranks governments across the planet according to the severity of their internet censorship, ranging from minimal or no censorship to very heavy censorship. Countries with moderate levels of internet censorship lie in between these two extremes. According Freedom House, only 13% of the world's people and a third of internet users live in countries with minimal censorship. On the other hand, roughly one-quarter of the world's population and internet users live under governments that engage in very heavy censorship, most notably China. As more people gain access to the internet worldwide, states are resorting to more complex forms of censorship such as firewalls and blocking/filtering websites while encouraging ISPs to police their users through self-censorship tactics, like in the case seen in Russia [9]–[11].

2.2 Censorship Methods

This section delves into the subject of internet censorship, examining the reasons behind it, the methods employed to censor online content, and the strategies used to bypass such censorship. It provides a comprehensive overview of the current state of internet censorship. From understanding the motivations behind censorship to exploring the technical methods used to achieve it, this chapter serves as a foundation for understanding the complexities of internet censorship and the ongoing efforts to fight against it.

2.2.1 DNS Manipulation

The Domain Name System (DNS) is a key component of the Internet. It provides a way for computers to locate websites on the Internet by translating human-readable domain names into numerical IP addresses that computers understand. DNS works by having authoritative name servers around the world that store

databases of IP addresses linked to domain names. When a user types in a website into their browser, their computer sends out a query to find the associated IP address for that website, and then connects to it directly [12].

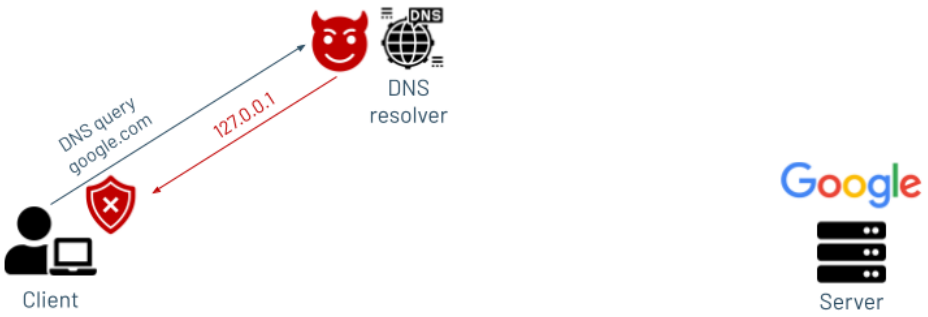


Figure 2.1: Visual representation of DNS manipulation.

Unfortunately, DNS can also be used as an effective tool for internet censorship by restricting access to certain domains or websites. This is done by blocking requests from users' computers or devices asking for specific domains or IPs, so they are unable to access them in most cases. In some countries this process has been implemented at an ISP (Internet Service Provider) level, where ISPs filter out requests related to certain domains and block users from accessing those websites. This type of restriction makes it very difficult for people in those countries to access censored content online as they are unable to connect with those sites even if they know their exact web address [7].

One of the most common method of DNS manipulation involves changing responses sent back from authoritative name servers when users try and look up domain information about blocked sites so that instead of connecting them with the actual website's IP address, it redirects them somewhere else (usually an error page). This makes it impossible for users trying to access censored information because even if they enter the correct web address into their browser, their computer will not be able connect with it since its looking up incorrect information due DNS tampering [12].

DNS can be used as an effective tool for internet censorship since it offers several ways of restricting user access and making sure certain types of content remain blocked from being accessed online regardless of what device or network people are using within censoring countries boundaries [12].

2.2.2 IP-Based Blocking

IP address based blocking is a technique used to block access to specific websites by preventing access to the IP address associated with the website. ISPs or states can implement IP blocking by maintaining a list of blocked IP addresses and configuring network devices, such as routers and firewalls, to block traffic to and from the IP addresses on the blocklist fig. 2.2.

Figure 2.2: Visual representation of IP-based blocking.

One common method for implementing IP blocking is to use firewalls to filter traffic at the network perimeter. ISPs or state-controlled entities can configure firewalls to block traffic to and from specific IP addresses, or to block traffic based on other criteria such as ports, protocols, or source and destination IP addresses. This method is often used by ISPs to block access to certain websites at the network level, by denying the traffic to reach the destination IP addresses [13].

Other ways of enforcing censorship by IP blocking is to use routers to filter traffic at the network edge. ISPs or state-controlled entities can configure routers to block traffic to and from specific IP addresses, or to block traffic based on other criteria such as ports, protocols, or source and destination IP addresses. This method is often used by organizations to block access to certain websites on their internal network [13].

2.2.3 HTTP(S) Interference

Internet censorship can be enforced on the HTTP and HTTPS protocols by using various techniques to inspect and filter network traffic. One common method is to use Deep Packet Inspection (DPI) to examine the contents of network packets and block specific types of content.

For the HTTP protocol, DPI can be used to inspect the application payload, such as the destination domain in the request headers, and block access to specific websites based on keywords or domain names. This can be done by maintaining a list of blocked keywords or domain names and using DPI to match the keywords

Figure 2.3: Visual representation of application layer blocking.

or domain names in the application payload with the items on the blocklist [14].

In the case of the HTTPS protocol, DPI focuses on the initial unencrypted message of a valid TLS `ClientHello` message, which contains the requested domain in the SNI (Server Name Indication) extension, providing a selector for censorship. Since the application payload of an HTTPS request is encrypted, it is not possible to inspect it directly, but by inspecting the SNI extension the censorship can be enforced [14].

2.2.4 Internet Shutdowns

An internet shutdown refers to the deliberate disruption of the internet or certain online services by a state or an ISP. This can be done by blocking or throttling specific IP addresses, disabling or jamming cellular networks, or by shutting down internet exchange points (IXPs) and internet service providers (ISPs). One common method for conducting an internet shutdown is to block or throttle specific IP addresses or IP ranges. This can be done by configuring routers or firewalls to filter traffic to and from specific IP addresses or IP ranges. This method is often used by ISPs or state-controlled entities to block access to specific websites or online services [15], [16].

A more locally enforced method is to disable or jam cellular networks. This can be done by shutting down cellular towers or jamming the radio frequencies used by cellular networks. This method is often used by state-controlled entities to disrupt communication and internet access in specific regions. It's important to note that internet shutdowns can have severe consequences, including the disruption of essential services such as healthcare and financial transactions, and can negatively impact human rights, such as freedom of expression, and economic activities. Moreover, internet shutdowns are often perceived as a sign of weakness by the government or ISP that conducts them, as they give the idea that they are not able to handle the dissent or information they don't want to be known [15], [17].

2.3 Censorship Circumvention

To circumvent censorship, various technologies have been developed, including Tor bridges, pluggable transport protocols, and Virtual Private Networks (VPNs). In this chapter, we will discuss the technical details of these technologies and how they can be used to bypass censorship measures and maintain online privacy.

2.3.1 Tor Bridges

Censors can use various techniques to block access to the Tor network. Some of the most common methods include blocking the IP addresses of known Tor relays, deep packet inspection (DPI) to identify and block Tor traffic, and using techniques like traffic fingerprinting and targeted scanning to identify and block access to Tor bridges. An example of a censor that has been known to block access to the Tor network is the Great Firewall of China (GFW). The GFW has been known to use a combination of techniques to block access to the Tor network. One of the main methods used is to block the IP addresses of known Tor relays. This is done by adding the IP addresses of known Tor relays to a blacklist, which is then used to block access to these IP addresses [18], [19].

Tor (The Onion Router) bridges are special types of Tor relays that are used to circumvent internet censorship. They are used to establish a connection to the Tor network in countries where the Tor network is blocked by the government. Tor bridges work by relaying traffic between users and the Tor network in a way that makes it difficult for censors to detect and block. Unlike regular Tor relays, bridges do not have a public IP address, and they are not listed in the public Tor directory. This makes it difficult for censors to locate and block them [20].

Tor bridges use a technique called pluggable transports to conceal the fact that a user is connecting to the Tor network. This technique can make the traffic between a user and a bridge appear as normal internet traffic, making it difficult for censors to distinguish it from other types of traffic. Bridges and the access to the Tor network provide anonymity to users who are at risk of being targeted by government surveillance or other forms of online tracking. This can be especially useful for human rights activists, journalists, and other individuals who need to protect their online privacy and security [21].

To use a Tor bridge, users need to obtain the bridge's address, which can be done through the official Tor website, by emailing the Tor Project [1], by Telegram or by using Moat which is built into the Tor Browser. Once a user has the bridge's address, they can configure their Tor client to use the bridge. Tor bridges are

[1] https://bridges.torproject.org/

an important tool for bypassing internet censorship and protecting online privacy. They work by relaying traffic between users and the Tor network in a way that makes it difficult for censors to detect, and they can be used to provide anonymity to users who are at risk of being targeted by government surveillance or other forms of online tracking [22].

2.3.2 Pluggable Transport

Pluggable transports (PTs) are a type of technology that can be used in conjunction with the Tor network to circumvent internet censorship. PTs are designed to disguise the fact that a user is using the Tor network, making it more difficult for censors to block or detect Tor traffic. PTs work by modifying the way that Tor traffic is transmitted over the internet. Instead of using the standard Tor protocol, PTs use a variety of different methods to conceal the fact that the traffic is associated with Tor. These methods include disguising Tor traffic as other types of traffic, such as HTTPS or SSH, or using encryption to conceal the contents of the traffic [23].

There are several different types of PTs that can be used with the Tor network, each with its own specific strengths and weaknesses. Some examples include obfs4, Scramblesuit, and meek. obfs4 is a PT that is able to quickly adapt to new censorship techniques and is particularly effective against deep packet inspection (DPI) based censorship. Scramblesuit is a PT that uses a combination of encryption, compression, and padding to conceal the contents of the traffic. meek is a PT that uses cloud providers to conceal the fact that the traffic is associated with Tor, making it more difficult for censors to block [24], [25].

Obfs4

The obfs4 protocol is a pluggable transport that is designed to disguise Tor traffic as regular internet traffic in order to evade internet censorship. It is based on the obfs3 protocol, but includes several improvements that make it more effective against censorship techniques such as DPI [25].

One of the main improvements of obfs4 is that it uses a more advanced encryption algorithm, called Elligator2. This encryption algorithm is designed to produce ciphertext that is indistinguishable from random noise, making it more difficult for censors to detect the traffic as being associated with Tor. Another improvement of obfs4 over obfs3 is that it uses a more advanced padding mechanism. Padding is the process of adding extra data to the traffic in order to make it appear as if the connection is being used for legitimate purposes. The obfs4 protocol uses a more advanced padding mechanism than obfs3, which makes it more difficult for censors to detect the traffic as being associated with Tor [25].

Additionally, Obfs4 includes a built-in mechanism for quickly adapting to new censorship techniques. The

protocol is designed to be resistant to DPI-based censorship, which is a common method used by censors to block Tor traffic. This is achieved by using a technique called "circuit scrambling" which is a way of changing the structure of the traffic periodically, making it more difficult for censors to detect and block the traffic [23].

Snowflake

Snowflake is a pluggable transport that is used by the Tor Project to circumvent internet censorship. It works as a proxy between Tor clients and the Tor network, making it more difficult for censors to block or detect Tor traffic.The Snowflake proxy is implemented as a browser extension that runs in the client's web browser. When a client wants to connect to the Tor network, it first connects to a Snowflake proxy. The proxy then establishes a connection to the Tor network on behalf of the client. This means that the client's traffic is first sent to the Snowflake proxy, and then forwarded to the Tor network. This makes it more difficult for censors to detect the traffic as being associated with Tor, as it appears to be regular web traffic [26].

It uses WebRTC, a technology that allows browsers to establish peer-to-peer connections, to establish the connection to the Tor network. This means that the Snowflake proxy and the Tor network can communicate directly, without going through a third-party server. This makes it more difficult for censors to block the connection. Additionally, a technique called "rendezvous" is used to establish the connection to the Tor network. During the connection handshake, the client and the Snowflake proxy exchange a series of public keys and a shared secret. These keys are used to encrypt the traffic that is sent over the connection. The encryption is designed to be indistinguishable from random noise, making it more difficult for censors to detect the traffic as being associated with Tor [26].

The Snowflake proxy is distributed through the Tor Project's website and can be easily installed by users as a browser extension. The Snowflake proxy is currently only available for the Chrome and Firefox browser [2].

Meek

Meek is a pluggable transport that is used by the Tor Project to circumvent internet censorship. It works by routing Tor traffic through a third-party server, typically a cloud provider such as Amazon Web Services (AWS), in order to conceal the fact that the traffic is associated with Tor. This makes it more difficult for censors to block or detect Tor traffic [27].

Meek uses a technique called "domain fronting" to conceal the fact that the traffic is associated with Tor. During the connection handshake, the client and the meek server exchange a series of public keys and a

[2] https://snowflake.torproject.org/

shared secret. These keys are used to encrypt the traffic that is sent over the connection. The encryption is designed to be indistinguishable from regular HTTPS traffic, which makes it more difficult for censors to detect the traffic as being associated with Tor. It uses the cloud provider's domain name as the front domain, which is the domain that is visible to censors. This makes it appear as if the traffic is regular HTTPS traffic going to the cloud provider's domain, rather than Tor traffic. The actual Tor traffic is sent as an HTTP request inside the HTTPS request, which is not visible to censors [27].

It's dedicated servers are typically operated by the Tor Project or other organizations that support internet freedom. They are distributed across multiple locations and can be easily accessed by clients through the Tor Browser. The Meek extension that is built into the Tor browser automatically chooses the best server based on network performance and censorship level of the client's location [27].

2.3.3 Shadowsocks

Shadowsocks is a proxy that is designed to bypass firewalls and other types of internet censorship. It is a free, open-source software that can be used to create a secure and private connection to the internet, by routing a user's internet traffic through a remote server, which acts as a proxy. The traffic is encrypted and decrypted at the client and server side respectively, making it more difficult for censors to detect and block the traffic. Shadowsocks uses a protocol called SOCKS5, which is a widely-used proxy protocol, to establish the connection between the client and the server [28].

The encryption used by shadowsocks is a stream cipher, which encrypts data one byte at a time. The encryption algorithm used is configurable, but the default one is AEAD (Authenticated Encryption with Associated Data) and it is based on the Advanced Encryption Standard (AES) algorithm. The encryption key is derived from a shared password, which is exchanged between the client and the server during the connection handshake [28].

When a user wants to connect to a Shadowsocks server, the client first establishes a connection to the server. During this connection, the client and the server exchange a series of public keys and a shared password. These keys are used to encrypt the traffic that is sent over the connection, and the shared password is used to derive the encryption key. Once the keys are exchanged, the client and server can establish the secure connection, and the user's internet traffic is routed through the Shadowsocks server, to its destination on the internet [28].

Shadowsocks is often used to circumvent internet censorship in countries where the government heavily monitors and blocks internet traffic. The encryption and SOCKS5 protocol makes it difficult for censors to detect and block Shadowsocks traffic, allowing users to access blocked websites and services. Additionally,

Shadowsocks can be used to conceal the user's location and browsing habits, making it more difficult for censors to track and block the user's internet traffic [28].

It is important to mention that while Shadowsocks is a powerful tool to bypass firewalls and internet censorship, it is not completely foolproof. Censorship techniques are constantly evolving and some governments have the resources to block Shadowsocks traffic. It is also important to use Shadowsocks over a secure network, as it can't protect against attacks on the local network level [28].

2.3.4 VPN

A Virtual Private Network (VPN) is a technology that allows users to establish a secure and private connection to the internet, by routing their internet traffic through a remote server. VPNs are commonly used to provide secure remote access to a company's internal network, but they can also be used to circumvent internet censorship [29].

VPNs work by creating an encrypted tunnel between the user's device and the VPN server. This tunnel is used to transmit all of the user's internet traffic, including web browsing, email, and instant messaging. The VPN server acts as a proxy, forwarding the user's internet traffic to its destination on the internet. Because the traffic is encrypted, it is more difficult for censors to block or detect [30].

They typically use a protocol called the Internet Protocol Security (IPsec) to establish the encrypted tunnel. IPsec is a set of security protocols that are used to establish secure connections over the internet, which provides confidentiality, integrity, and authenticity for the transmitted data.When a user wants to connect to a VPN, they first establish a connection to the VPN server. This connection is typically established using the Internet Key Exchange (IKE) protocol. IKE is a protocol used to establish a secure connection between two devices. It is used to establish the shared secret keys that are used to encrypt and decrypt the VPN traffic. Once the connection to the VPN server is established, the user's device and the VPN server use the IKE protocol to negotiate the shared secret keys that will be used to encrypt and decrypt the VPN traffic. Once the shared secret keys are established, the user's device and the VPN server use the IPsec protocol to establish the encrypted tunnel [29], [30].

VPNs can be used to circumvent internet censorship by routing a user's internet traffic through a VPN server that is located in a country where the censorship is less severe. This can allow users to access websites and services that are blocked in their own country. Additionally, VPNs can be used to conceal the user's location and browsing habits, making it more difficult for censors to track and block the user's internet traffic [31].

2.4 Active Probing

The Great Firewall of China (GFW) is a system of internet censorship and surveillance used by the Chinese government to regulate and monitor internet access in the country. One of the methods used by the GFW to discover and block circumvention servers is an active probing technique [19].

Active probing is a technique used to identify hidden networks, such as Tor bridges. It involves sending out special requests from a server located in the GFW network to computers outside the network, with the intention of detecting and blocking any computers responding to the requests. This is done by sending out a series of requests to different servers, each of which has a specific set of parameters set. The requests are sent out in a specific order, allowing the GFW to identify and block any servers responding to the requests [32].

The active probing technique used by the GFW is designed to detect and block Tor bridges, which are servers that allow users to access the Tor network. Tor bridges are designed to be hidden, so that they cannot be easily detected by the GFW. The active probing technique works by sending out requests to different servers, each with a specific set of parameters set. The requests are sent out in a specific order, and if any of the servers respond with the specific parameters set in the request, then it is identified as a Tor bridge [19].

Once a server is identified as a Tor bridge, the GFW can block it by using a variety of techniques such as IP blocking, packet filtering, and URL filtering. The GFW also uses other techniques such as deep packet inspection to detect and block circumvention servers, but the active probing technique is one of the most effective methods of detecting and blocking Tor bridges [19].

The active probing technique has been used successfully by the GFW to detect and block Tor bridges, and it is an effective way of monitoring and controlling internet access in China. It is also a useful tool for other countries, as it can be used to detect and block other types of circumvention servers [19].

2.5 HTTPT - A Probe Resistant Proxy

HTTPT aims to provide better protection against censorship and active probing attacks than existing designs. One of the main benefits of HTTPT is that it is immune to replay attacks, which are a common problem for existing probe-resistant proxies. This is achieved by using the Transport Layer Security (TLS) protocol, which includes bidirectional nonces in the handshake [33].

Another benefit of HTTPT is that it leverages existing web servers, making it more difficult for censors to single out or identify. This obviates the need to mimic TLS or other server fingerprints that could allow censors to block it. Additionally, HTTPT has minimal overhead after the initial TLS handshake, which is achieved by using WebSockets between the client and server, which allows for the efficient transfer of binary

data [33].

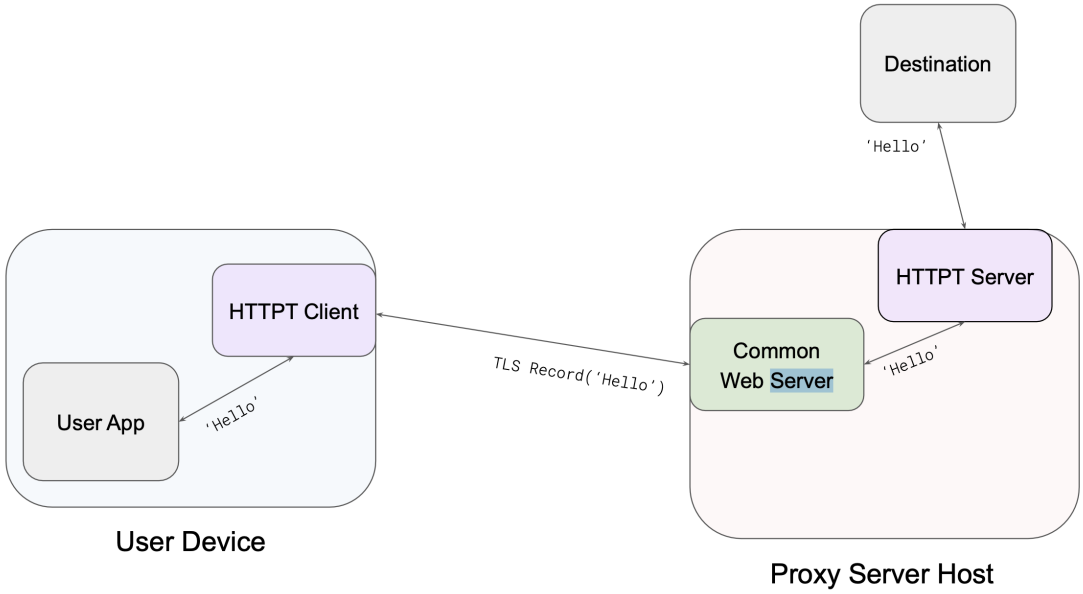


Figure 2.4: Overview of the HTTPT sequence [33].

HTTPT design includes several features that are intended to make it difficult for censors to identify and block the proxy. These include using existing popular TLS implementations and servers, avoiding identifying characteristics in the TLS layer, and responding with realistic certificates. However, the contents of the payload sent by the HTTPT server must be carefully chosen so that censors cannot identify it based on its responses. One approach proposed is to place HTTPT proxies at already existing HTTPS servers run by volunteers, which already have valid certificates and content that can be returned to a probing censor [33].

Alternatively, HTTPT could mimic existing servers not controlled by the HTTPT proxy admin. In that case, HTTPT would transparently proxy all traffic to a dedicated “mask site”, which is an external TLS server. When a censor makes a connection to the HTTPT server, their packets are relayed to the mask site, and the responses are sent back to the censor, making it appear as if the HTTPT server is the mask site. The HTTPT proxy can detect the secret from the client, in such way that censors unable to prove knowledge of the secret will continue to have their traffic transparently relayed to the mask site, making it difficult for the censor to detect [33].

The HTTPT proxy was the inspiration behind this thesis, since it presents a novel approach of leveraging web servers as proxies. The goal of this work is to provide approaches like HTTPT, with an application that can automate the process of generating authentic content and therefore ease the barrier for internet freedom activists to setup their own server instances for internet censorship circumvention.

2.6 Generating Synthetic Content

This section provides an overview of the background knowledge on Natural Language Processing, Machine Learning and state-of-the-art models such as Transformer models, Artificial Neural Networks and Language Models. It will discuss ML concepts and algorithms, such as supervised and unsupervised learning, and the role they play in NLP. The chapter will also delve into the Transformer model architecture, its improvements over previous models, and how it has been applied to various NLP tasks. Furthermore, it will also cover Artificial Neural Networks, a key component of transformer models and how they are used to model complex data. Finally, the chapter will give an overview of the powerful language model GPT-3 and its capabilities in various NLP tasks.

2.6.1 Machine Learning

Machine Learning is a subfield of Artificial Intelligence that enables computer systems to learn from data, without being explicitly programmed. It is a method of teaching computers to recognize patterns and make predictions or decisions. Machine learning algorithms can be broadly categorized into three types [34]:

1. Supervised learning: the algorithm is trained on a labeled dataset, where the correct output is provided for each input. The algorithm learns to make predictions based on the input-output pairs. Examples include linear regression, logistic regression, and decision trees.
2. Unsupervised learning: the algorithm is trained on an unlabeled dataset and must find the underlying structure or pattern in the data on its own. Examples include k-means clustering, principal component analysis, and autoencoders.
3. Reinforcement learning: the algorithm learns by interacting with the environment and receiving feedback in the form of rewards or penalties. The goal is to learn a policy that maximizes the cumulative reward over time.

The technical details behind machine learning include various mathematical and computational concepts such as probability and statistics, linear algebra, optimization and gradient descent, and neural networks. The use cases for machine learning are vast and cover credit scoring, fraud detection, image and speech recognition, customer segmentation and natural language processing. It is also used in natural language processing, computer vision, speech recognition and many other areas. The increasing availability of data and advances in computational power have made it possible to train more sophisticated models and apply machine learning in an increasing number of areas [34], [35].

2.6.2 Natural Language Processing

Natural Language Processing (NLP) is a field of Artificial Intelligence that deals with the interaction between computer systems and human languages, in text or speech format. The goal of NLP is to enable computers to understand, interpret and generate human language, making it possible to interact with machines in a way that is more natural for humans [36].

NLP relies on a combination of linguistics, computer science, and machine learning to analyze and understand human language. It involves several sub-tasks such as:

- Text pre-processing: cleaning and preparing the text data, such as lowercasing, tokenization, stemming, and stop-word removal.
- Semantic parsing: analyzing the meaning of a sentence, such as identifying the subject, predicate, and objects.
- Named entity recognition: identifying and classifying entities mentioned in the text, such as people, organizations, and locations.
- Sentiment analysis: determining the sentiment (positive, negative, neutral) expressed in a text.
- Text summarization: reducing a text to its most important information.
- Text generation: creating new text that is coherent and similar to a given input text, such as language translation, text summarization and question answering.
- Text-to-speech: converting text into speech, such as text-to-speech synthesis and dialogue systems.

NLP covers various algorithms and techniques such as rule-based methods, statistical methods, and machine learning methods. Rule-based methods use a set of predefined rules and patterns to analyze text, while statistical methods use statistical models to analyze text data. Machine learning methods, on the other hand, use algorithms such as decision trees, support vector machines, and neural networks to train models on large amounts of data and make predictions [36], [37].

NLP has a wide range of use cases, from customer service chatbots and virtual assistants to text-based search engines and sentiment analysis for social media monitoring. NLP is also used in language translation, language generation, summarization, question answering, and many other tasks [36], [37].

2.6.3 Artificial Neural Networks

Artificial Neural Networks (ANNs) are a type of machine learning model inspired by the structure and function of biological neural networks in the human brain. They are made up of layers of interconnected nodes or "neurons," which process and transmit information. Each neuron in an ANN receives input from

other neurons, performs a simple computation on that input, and then sends the result to other neurons in the next layer. This process continues until the final output is produced 2.5. The computations performed by the neurons are determined by a set of parameters, called weights and biases, that are learned during the training process [38].

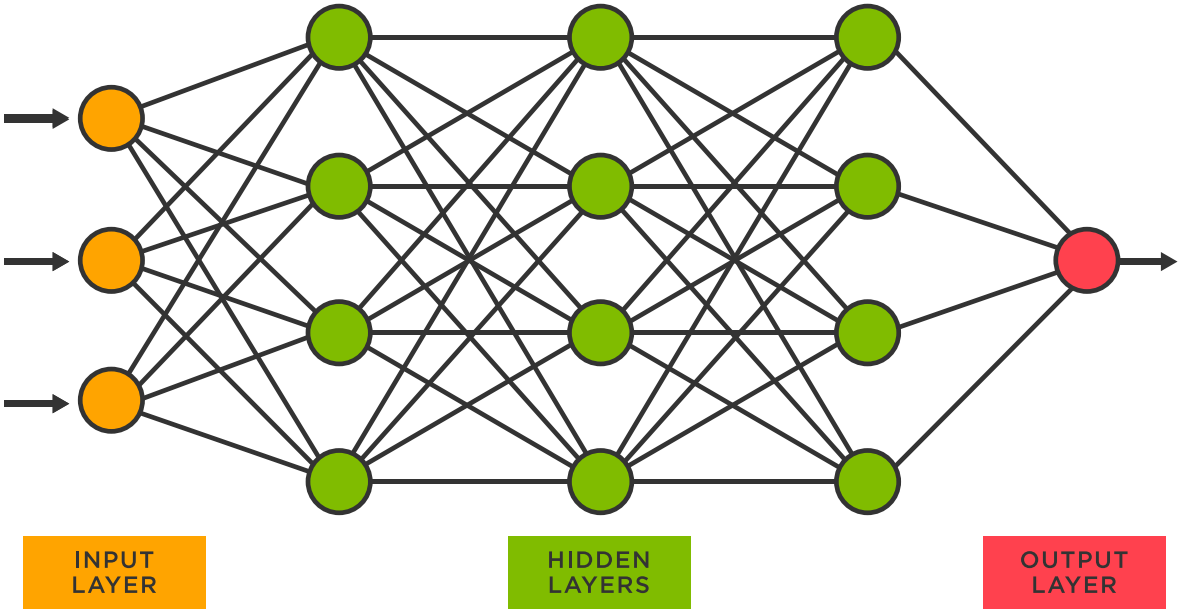


Figure 2.5: Visual depiction of an ANN [39]

The training process of an ANN involves using a dataset of input-output pairs to adjust the weights and biases of the network so that it can accurately predict the output for a given input. This is typically done using an optimization algorithm such as gradient descent, which adjusts the weights and biases in the direction that reduces the error between the network's predictions and the true outputs [38].

Feedforward neural networks are the simplest type of ANNs. They are composed of layers of neurons that are fully connected, meaning that each neuron in one layer is connected to every neuron in the next layer. FFNNs are used for a variety of tasks such as image classification, speech recognition and natural language processing [40].

Convolutional neural networks are a type of ANN that is specifically designed for image recognition tasks. They consist of layers of neurons that are organized in a way that preserves the spatial structure of the input. CNNs are particularly effective at identifying patterns and features in images, such as edges and shapes [40].

Recurrent neural networks are designed to process sequential data, such as time series, speech and text. They have a feedback connection that allows information to flow through the network multiple times, which allows the network to maintain a "memory" of previous inputs [40].

Artificial Neural Networks are used for image and speech recognition, natural language processing, predictive maintenance, anomaly detection [40].

2.6.4 Transformer models

Transformer models are a type of computer program that are designed to understand and generate human-like text, they are particularly useful in the field of NLP. The main concept behind transformer models is that they process text in a more efficient way than traditional methods by analyzing the whole sentence at once, rather than one word at a time. This allows the model to understand the context of the text and make predictions or decisions based on that understanding [41], [42].

The transformer model architecture is made up of two main parts: an encoder and a decoder. The encoder reads the input text and creates a set of features that represent the meaning of the text. The decoder then uses these features to generate the output text. The key component of the transformer model is the self-attention mechanism, which allows the model to weigh the importance of different parts of the input text when producing the features [41], [42].

An encoder is a neural network architecture that processes the input text and produces a set of features that represent the meaning of the text. The encoder typically consists of a stack of layers, each layer consisting of a set of self-attention and feed-forward neural network sublayers. The self-attention sublayer allows the encoder to weigh the importance of different parts of the input text when producing the features. It does this by computing a set of attention weights, which indicate how much each part of the input text should be considered when generating the features. The attention weights are computed using a set of learnable parameters called the attention weights. The feed-forward sublayer is a simple neural network architecture that is used to further process the features generated by the self-attention sublayer. It consists of a linear transformation followed by a non-linear activation function, such as a rectified linear unit. The encoder layers are typically arranged in a stack and the output of each layer is passed as input to the next layer 2.6. This allows the encoder to capture more context and fine-grained information about the input text as the information flows through the layers [41]–[43].

A decoder takes the features produced by the encoder and generates the output text. The decoder also typically consists of a stack of layers, each layer consisting of a set of self-attention and feed-forward neural network sublayers. The self-attention sublayer in the decoder works similar to the self-attention sublayer in the encoder. It allows the decoder to weigh the importance of different parts of the input features when generating the output text. The attention weights are computed using a set of learnable parameters, similar to the encoder. The feed-forward sublayer in the decoder is also similar to the feed-forward sublayer in the

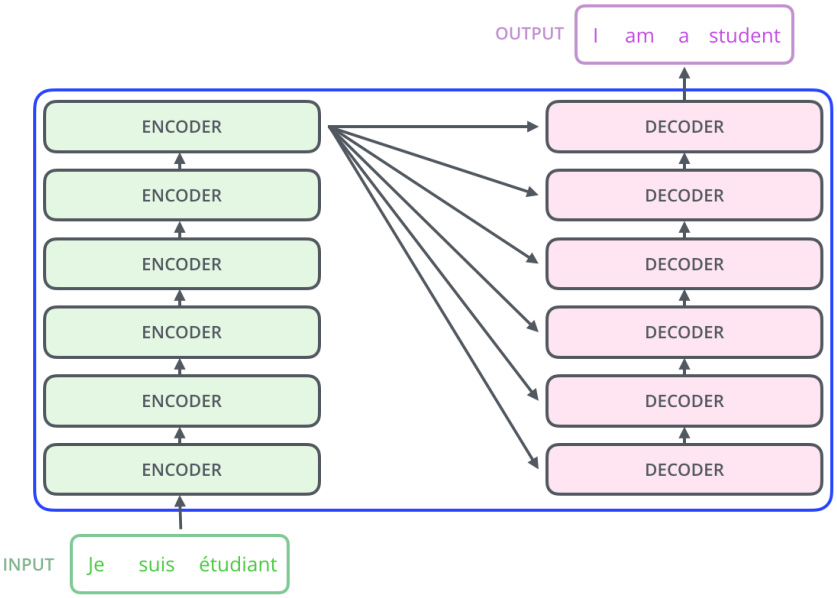


Figure 2.6: Illustration of the encoder and decoder process [44].

encoder, it's a simple neural network architecture that is used to further process the features generated by the self-attention sublayer. The decoder layers are also typically arranged in a stack, similar to the encoder layers 2.6, and the output of each layer is passed as input to the next layer. This allows the decoder to capture more context and fine-grained information about the input features as the information flows through the layers [41]–[43].

2.6.5 Language Models

A language model is a type of machine learning model that is trained to predict the probability distribution of a sequence of words in a given text. The concept behind a language model is to train a computer program to predict the next word in a sentence, given the previous words. The program learns to understand the structure and the meaning of the text by analyzing a large corpus of text data. This is done by estimating the probability of a given word, given the sequence of previous words. Once the model is trained, it can be used for a variety of natural language processing tasks, such as language translation, text summarization, and text generation [45], [46].

Language models can be divided into two main categories: generative models and discriminative models. Generative models are trained to generate new text that is similar to the training data, by learning the probability distribution of the text. They can be further divided into non-parametric and parametric generative models. Non-parametric models like n-gram models estimate the probability of a word based on its preced-

ing words, while parametric models like RNNs and transformer models estimate the probability of a word based on a more complex function learned from data. Discriminative models, on the other hand, are trained to classify the text into a specific category or label. The main difference between generative and discriminative models is that the former models the probability distribution of the data while the latter models the decision boundary between classes [45], [46].

The architecture of a language model can vary, traditional models are based on RNNs which process the text in a sequential way, one word at a time. However, recent models like GPT-3 are based on transformer models, which process the text in a parallel way, allowing the model to handle longer input sequences and capture more context. This is done by using a self-attention mechanism that allows the model to weigh the importance of different parts of the input text when generating the output. This self-attention mechanism is based on computing attention weights, which indicate how much each part of the input text should be considered when generating the features. These attention weights are computed using a set of learnable parameters called the attention weights [45], [46].

2.6.6 GPT-3

GPT stands for "Generative Pre-trained Transformer." It is a type of language model developed by OpenAI. GPT is a generative model, which means that it generates text from scratch, as opposed to a discriminative model, which would classify text into categories. It is pre-trained, meaning that it was trained on a large dataset of text before it was fine-tuned for specific tasks. The pre-training allows the model to learn general language patterns and structures that are useful for a wide range of natural language processing tasks, such as language translation, text summarization, and question answering [41], [46]–[48].

The training process for GPT involves feeding the model a large dataset of text and having it predict the next word in the sequence. This is known as unsupervised learning, which means that the model is not provided with explicit labels or tasks during training. Instead, it learns to generate text by analyzing patterns in the data. The training data for GPT is typically a large corpus of text, such as books, articles, or websites. The model is trained on this text by processing it word by word, and trying to predict the next word in the sequence. For example, if the model is trained on the sentence "The cat sat on the", it would try to predict the next word, which is "mat" [41], [47], [48].

The training is done using a variant of the transformer architecture called the transformer decoder, which is an attention-based neural network that is able to handle long-term dependencies in the text. The transformer decoder is composed of multiple layers of attention and feed-forward neural networks, which are trained to extract features from the input text and use them to generate the next word in the sequence. Additionally,

maximum likelihood estimation is being used, which is a common method used in unsupervised learning. This method involves adjusting the model's parameters so as to maximize the likelihood of the training data given the model. In other words, it adjusts the model's parameters so that the model's predictions are as close as possible to the actual training data.As the training process progresses, the model becomes better at predicting the next word in the sequence, and it starts to learn general patterns and structures in the text. After training, the model can be fine-tuned for specific tasks using a smaller dataset and it can be used to generate text, translate text, answer questions, summarize text, and more [46]–[48].

GPT-3 is one of the latest version of GPT, which is trained on a massive dataset of 570GB of text data, and it is trained using a technique called unsupervised learning, which means that it was not provided with any explicit task to perform during training. Instead, it learned to generate text by analyzing patterns in the data. GPT-3 can be fine-tuned for specific tasks using a smaller dataset, and it can be used to generate text, translate text, answer questions, summarize text and more. It can also be used to generate code, complete tasks, and answer question with high level of accuracy [48].

3 Related Work

The paper "Geneva: Evolving Censorship Evasion Strategies" Bock *et al.* introduces a genetic algorithm, called Geneva, that automatically discovers censorship evasion strategies by combining primitive operations in various ways and evaluating the combinations against a network censor, whether real or simulated. The discovered strategies are packet-level manipulations similar to those found in prior work such as Bock *et al.* [49]. Geneva was trained and evaluated in the lab against simulated censors and in the wild against real censors in China, India, and Kazakhstan.An evasion strategy in Geneva consists of paired triggers and actions, where a trigger is a predicate over packets and an action is an operation on a single packet. These actions include duplicating, fragmenting, and tampering with packets, and can be organized into a tree structure. The fitness of an individual strategy is determined by its effectiveness against the censor with penalties for large action trees or high network overhead. The authors report that each new generation of strategies takes 5-10 minutes to compute and full training takes 4-8 hours. The bulk of the authors' validation was in China against the Great Firewall, where Geneva found a number of strategies that confuse the firewall's notion of the correct TCP sequence number or connection status. The paper also reported on some unexpected strategies that seem to expose previously unknown characteristics of the GFW's classification algorithms. The authors also tested in India and Kazakhstan where they found effective strategies that were comparatively simpler than the ones found in China.

Kaptchuk *et al.* [50] presents a symmetric-key steganography protocol that allows for the exchange of messages that conform to a certain text generation model while also encoding a hidden message. The protocol, called Meteor, uses sophisticated generative models of natural language text, such as GPT-2, to provide realistic covertexts. Meteor adapts its encoding rate based on the local entropy of the text generation model, unlike past schemes that can fail when entropy is low. The authors of the paper demonstrate how Meteor can be used in practice by detailing the protocol's method of encoding and decoding messages using a probability distribution of candidate words, a pseudorandom bitstream, and a reverse lookup in the probability table. The paper highlights the advantage of Meteor's natural adaptation to varying entropy levels and the ability for the sender and receiver to know how many bits have been decoded from a given word.

Meek's Susceptibility to Classification based on Traffic Flow Analysis by Sheffey *et al.* [51] examines the potential for traffic flow analysis, specifically packet sizes and packet timing, to be used in identifying the use of meek, a censorship circumvention tool. The authors collect their own traffic traces of browsing home pages with and without meek-with-Tor, and use this data to develop classifiers that can distinguish ordinary HTTPS from meek HTTPS. They then demonstrate how minimal perturbation of the meek-derived feature vectors can hinder the classifiers. To build a corpus of training and test data, the authors used a parallel data collection framework using Docker containers and a centralized work queue. They browsed 10,000 home pages using both a headless Firefox and Tor Browser configured to use its meek-azure bridge from three different networks, yielding a total of 60,000 traffic traces. From these traces, the authors extracted binned features including TCP payload length and interarrival times, tagged with direction. The authors use a GAN (generative adversarial network), specifically the StarGAN implementation, to iteratively transform a meek feature vector so that it looks more like an ordinary HTTPS feature vector. The transformation process tries to minimize the size of changes required by including a perturbation loss term that increases as more changes are required. The goal of minimizing perturbation is to make it easier to implement the resulting distribution while still fooling the classifiers.

VanderSloot *et al.* [52] describe the development a system for measuring application-layer censorship using the echo protocol (TCP port 7, RFC 862). The system, called Quack, works by sending messages (such as HTTP requests or TLS ClientHello with SNI) to an echo server, and then checking the responses. The authors also account for potential complications, such as repeating the test multiple times and testing both likely censored and uncensored messages against the same echo server. The paper compares Quack to other existing remote measurement systems, such as Augur for TCP/IP and Satellite and Iris for DNS. The authors present results from a worldwide test of Quack, which included a ZMap scan to identify hosts with port 7 open, testing URLs from multiple sources including the Citizen Lab test list and the Alexa top 100K, and targeting discard servers (port 9, RFC 863) to determine if censorship is taking place through inbound or outbound blocking. The results of the test identified blocking in 13 countries, including China, Egypt, Iran, and Turkey, and revealed notable differences in blocking proportions across HTTP and TLS. The authors also noted the absence of certain known censors, such as Russia and Pakistan, suggesting the use of alternative methods such as DNS poisoning.

Alharbi *et al.*'s paper [53] is the first systematic and longitudinal study of digital filtering in Saudi Arabia. The authors focus on the filtering of web sites (the Alexa top 500 in 18 content categories) and social media/communications apps (18 apps such as iMessage, Line, and WeChat). The study was conducted over a period of three years, between March 2018 and April 2020, from vantage points in four cities and three

ISPs. The study found that overall, there has been a trend towards less filtering, with a moderate decline in the blocking of web sites and a marked decline in the filtering of mobile apps. The authors used a variety of methods to test for filtering, including DNS lookups over UDP and TCP, direct TCP connections to port 80, HTTP requests directly to the target web server, and TLS connections directly to the target web server. The study found that web site filtering is based on HTTP and TLS features, not DNS or IP address. The study also found that for mobile apps, those that support voice and video calling have gone from being almost completely filtered to almost completely unfiltered, which corresponds to a deliberate relaxation of policy towards communication-oriented apps that occurred in 2017. The authors also consulted with a government official and expert in local law, who stated that the study did not violate the law. The study provides a detailed picture of the state of digital filtering in Saudi Arabia, including the specific methods used for filtering and the changes that have occurred over the study period.

4 Approach

4.1 Keyword Selection For Synthetic Content

This chapter presents the results of an experiment aimed at identifying which keywords are the most susceptible to Internet censorship. It was conducted by first identifying infrastructural web servers, which was done by gathering IP and meta information. A remote measurement for censorship was then conducted on these identified servers. The collected data was analyzed to determine which keywords were sensitive to internet censorship. The findings of this experiment provide valuable insights into the current state of internet censorship and the keywords that are most at risk of censorship, thus providing a basis for the development of the Python application presented in the second experiment.

4.1.1 Methodology

In order to determine what kind of content is susceptible to being censored, an experiment was carried out for this thesis. To test country by country, which content is currently being blocked from being accessed through the HTTP and HTTPS protocol, without setting up testing machines in each country worldwide, a remote measurement technique, similar to the concept of Hyperquack [54] has been used.

The concept is based on predictable errors that can indicate if a censor has interfered with the communication. To illustrate the concept of predictable errors, fig. 4.1 displays a successful flow of HTTPS network communication. The machine obtained a valid IP address for google.com and was able to establish a connection to a web server hosting Google's search engine.

However, if the measurement machine established a connection to a different web server that doesn't host google.com, it will answer with a predictable error as shown in fig. 4.2.

This behavior can be used to detect Internet censorship remotely as depicted in fig. 4.3. The measurement machine send multiple control requests for domains that either don't exist at all or simply not hosted on the web server targeted for testing, like example1.com, example2.com and so on. The web server will reply to all the requests with the same errors message, in this case with a status code 302, since the requested

Figure 4.1: Establishing a succesful connection to google.com.

website is unknown to the server. By doing so a solid predictable response template can be created for that web server.

This is crucial for the final step, in which different keywords, or more specifically in the case of HTTPS, different domain names, will be sent to the tested web serer. Let's assume that access to Facebook is being blocked in a specific country. If a censor observing the network traffic is placed somewhere in the path between the measurement machine and the web server, certain keywords, in this case facebook.com, will cause the censorship device to interrupt the traffic flow by either injecting a RST-packet within the TCP flow, thereby ending the communication or by dropping the packet all together.

Other methods include blockpages that will be injected or a redirection to a different machine where the blockpage will be fetched from, notifying the user that the access of the website is restricted. Either one of those scenarios will generate a response that deviates from the template generated by the control measurements and therefore reveal which keywords are being censored within the country the web server is being hosted in.

4.1.2 Selecting Web Servers

In order to carry out this large scale network scan, access to the server machines from the organization Censored Planet[1] was provided. This allowed a scan of this magnitude without being interrupted by an upstream ISP. Additionally, the ethical scanning guidelines developed by ZMAP [55] were being complied

[1] www.censoredplanet.org

Figure 4.2: Generating a predictable error response.

with, which for example provides practices to minimize the risk of scanning machines from individuals, since these forms of worldwide scans could cause web servers to crash, or let individuals or other organizations falsely assume that they are being attacked. To comply with these guidelines, only IP addresses from infrastructural web servers were selected for the experiment from the following sources:

- APNIC [56]: The regional internet registry administering IP addresses for the Asia Pacific.
- CAIDA [57]: Center for Applied Internet Data Analysis, conducts network research and builds research infrastructure to support large-scale data collection, curation, and data distribution to the scientific research community.
- Maxmind [58]: Database that provides country information for IP addresses.
- PeeringDB [59]: A freely available and user-maintained database of networks.
- Censys [60]: A web-based search platform for assessing attack surface for Internet connected devices, which offers a large database for finding and identifying web servers worldwide.

Since the internet is a network of networks, where each network is being referred to as autonomous system (AS), services like aspop [2] from APNIC can be used to get valuable inside within the landscape. As of December 28, 2022, the service recorded 28.978 different AS information, and additionally estimates the amount of users served per AS organization.

This is valuable information, since the goal of the experiment is to identify those networks that serve the

[2] https://stats.labs.apnic.net/aspop

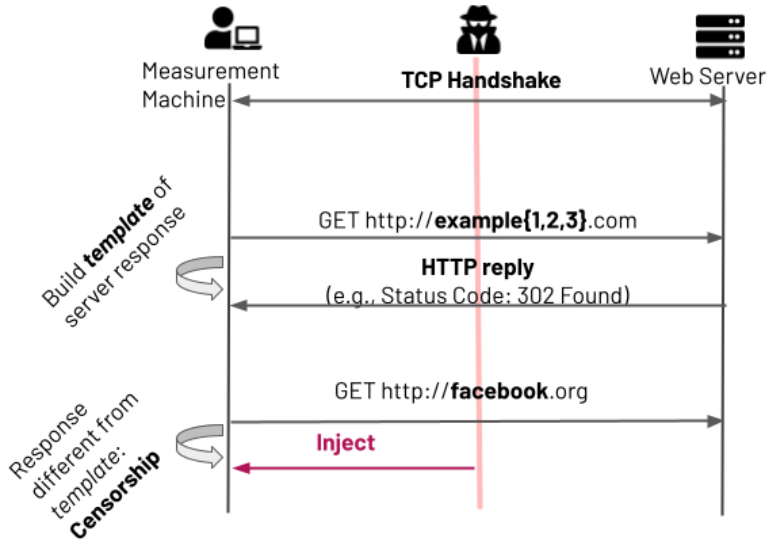


Figure 4.3: Censorship device injecting and therefore blocking the connection.

most users within a given country and helps to avoid organizational networks, since they tend to be more restrictive on what websites can be accessed by their employees and wouldn't serve as a valid representation of the censorship landscape faced by regular users.

```
1  const ApnicURL = "https://stats.labs.apnic.net/aspop"
2
3  func fetchApnicData() []ApnicRow {
4      log.Println("Fetching data from APNIC")
5      resp, err := http.Get(ApnicURL)
6      if err != nil {
7          log.Println(err)
8      }
9      body, err := ioutil.ReadAll(resp.Body)
10     if err != nil {
11         log.Println(err)
12     }
13
14     content := string(body)
15     splitData := regexp.MustCompile("(?m:^\\s*\\[.*$\\n)")
16     removeAddressCommas := regexp.MustCompile(
17         "(?m:\"[^\",]+),([^\"]+\")")
18     content = removeAddressCommas.ReplaceAllString(content, "")
19     cleaned := splitData.FindAllString(content, -1)
```

```
20      var rows []ApnicRow
21
22      for _, v := range cleaned[1:] {
23          v = strings.TrimSpace(v)
24          replacer := strings.NewReplacer("[", "", "]", "", "\"", "")
25          v = replacer.Replace(v)
26          a := strings.Split(v, ",")
27          var percent float64
28          if percent, err = strconv.ParseFloat(a[5], 32); err != nil {
29              log.Println(err)
30          }
31          rows = append(rows, ApnicRow{As: a[1][2:],
32              Country: a[3][len(a[3])-6 : len(a[3])-4],
33               Percent: float32(percent)})
34      }
35
36      return rows
37  }
```

Listing 4.1: Fetches and parses aspop data

However, APNIC doesn't offer a dedicated API to fetch the data, this is why an algorithm 4.1 was developed to parse the HTML page for the information needed.

Luckily, CAIDA provides a GraphQL interface to fetch organizational information from their database. It ranks AS networks by their customer cone size, which is the number of direct and indirect customers served. This metric will be used in combination with the data provided by APNIC.

APNIC's data provides a metric that determines what percentage of the population each AS in a given country serves, CAIDA's customer cone will be used to get estimation of how big a certain AS is based on it's IPv4 address space and can therefore be used to limit on how many IP addresses should be selected for each AS.

While the two services mentioned before are solely used to get information on the actual AS networks within every country worldwide, PeeringDB's database will be (ab)used to identify infrastructural web servers, since it records the websites of each AS organization registered, valuable information that is missing in the CAIDA database. These domains are commonly hosted by the AS organization themselves and are therefore ideal targets for the experiment. The script written for retrieving the desired information uses ZDNS for resolving the domain names to IP addresses, resulting in a total of 20.974 addresses.

The source of the second batch of possible web servers was obtained by Censys databases and included

1.749.057 IP addresses of ethical servers, resulting in a total number of 1.770.031 infrastructural web servers.

The developed algorithm 4.2 takes all the obtained AS information, selects a minimum of 10 AS networks and adds additional networks to it until a coverage of at least 95% per country is being reached, while limiting the number of IP addresses per with a maximum of 60. The country information for the IP addresses has been obtained using Maxmind.

```
1  func SelectIps(input []string, selectedAS map[string][]string,
2      coneSizes map[string]int, countryDbPath string,
3      asnDbPath string) []string {
4      // Selects the actual IPs
5      ipv4Mask := net.CIDRMask(24, 32)
6      countryDb, err := geoip2.Open(countryDbPath)
7      if err != nil {
8          log.Panic(err)
9      }
10     asnDb, err := geoip2.Open(asnDbPath)
11     if err != nil {
12         log.Panic(err)
13     }
14
15     defer countryDb.Close()
16     defer asnDb.Close()
17     count := make(map[string]int)
18     subNetCount := make(map[string]int)
19     var ips []string
20     // Setting the limits for each country AS based on the ipLimit
21     for country, selectedAS := range selectedAS {
22         for _, as := range selectedAS {
23             key := fmt.Sprint(country, as)
24             count[key] = ipLimit(coneSizes[as])
25         }
26     }
27     for _, s := range input {
28         ip := net.ParseIP(s)
29         countryRecord, errCountry := countryDb.Country(ip)
30         asRecord, errAs := asnDb.ASN(ip)
31         if errCountry != nil || errAs != nil {
32             continue
```

```
33          }
34          kkey := fmt.Sprint(countryRecord.Country.IsoCode,
35              asRecord.AutonomousSystemNumber)
36          if count[key] > 0 {
37              subNet := ip.Mask(ipv4Mask).String()
38              if subNetCount[subNet] < config.MaxSubnet {
39                  subNetCount[subNet]++
40                  count[key]--
41                  ips = append(ips, s)
42              }
43          }
44      }
45      return ips
46  }
```

Listing 4.2: Combining all gathered metrics to determine the final set of web servers

After the selection filtering process has been applied a total of 13.088 web servers, covering 182 countries and 1.082 AS networks have been selected.

4.1.3 Results

The experiment was conducted on December 14, 2022, and it took approximately 36 hours to finish the scan, resulting in a total of 17.605.448 data points for the HTTPS and 25.953.881 for the HTTP scan.
The list of domains to test was sourced from the Citzien Lab's test list [61], which contains URLs for the specific purpose to discover censorship worldwide and the Tranco list [62], which itself includes 1.000.000 million domains, of which only a subset was used, resulting in a total number 868.672 of domains used for the experiment. A database for categorizing the domains tested was provided by Censored Planet.
For each domain tested on a infrastructural web server, multiple probes with random delays of a few seconds in between, have been sent to account for possible networks errors. Starting with the control measurements that were used to generate an expected error template from the tested servers4.3.

```
1  func (h *HttpProtocol) GenerateTemplate(response interface{},
2      keyword string) (
3      template interface{}, err error) {
4      httpResponse, ok := response.(*HttpResponse)
5      if !ok {
6          return nil, errors.New("Expected response HttpResponse")
```

```
7       }
8       filterDomain := newDomainFilter(keyword)
9       filterBody := func(body string) string {
10          body = timestampRegex.ReplaceAllString(body,
11              TimestampReplacmentMarker)
12          return filterDomain(body)
13      }
14      return &HttpTemplate{
15          StatusLine: filterDomain(httpResponse.StatusLine),
16          Headers:    httpResponse.Headers,
17          Body:       filterBody(httpResponse.Body),
18      }, nil
19  }
```

Listing 4.3: Combining all gathered metrics to determine the final set of web servers

A multi threaded Golang application was developed to send out the probes, receive and parse the response into an JSON file. Before the response was saved into a custom struct type, meta information about the probed server were added using the Maxmind database.
Each data entry contained the following fields:

- measurement-id: Each probe was assigned with an unique ID for identification.
- domain: Specifies for which domain the web server has been probed for.
- domain-category: Specifies the category of the tested domain.
- control-domain: Indicates if the domain was used for creating the expected response template.
- start-time: Start of the probe measurement.
- end-time: End of the probe measurement.
- server-ip: IP address of the tested server.
- server-asn: AS number of the network in which the server resides in.
- server-country: Country in which the server resides in.
- error: Error message received from the probe.
- tls-version: TLS version offered by the server.
- tls-cipher-suite: Cipher suites offered by the server.
- certificate: Server certifcate.
- certificate-matches-domain: Boolean value to indicate if the certificate matches the probed domain name.

- certificate-name: Common name issued within the certificate.
- certificate-issuer: Issuer of the server certificate.
- status: Status code of the response.
- http-headers: HTTP header information of the response.
- received-body: HTTP body information of the response.
- is-known-blockpage: True if the blockpages signature matches a known one.
- page-signature: Fingerprint of the blockpage.
- no-response-matches: True if all responses of the trial don't match the expected template.
- control-failed: True, if the control probe failed.

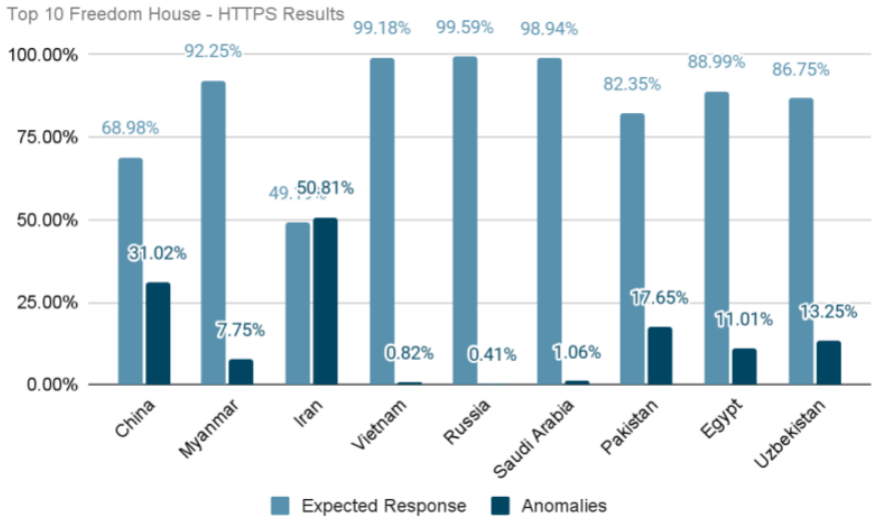


Figure 4.4: Overview of matching and mismatching responses within the HTTPS result set.

Multiple Python scripts were created for the analysis of the resulting JSON file. While one half of the fields are used to provide meta information about the server, the second was used to analyze the response generated and create appropriate outcome classifications.

Before the final analysis was conducted, the data was vetted to exclude responses were the initial control measurement failed, thereby excluding all results for these kind of web servers.

To provide a glimpse into the massive data set collected, the focus will be set on the most heavily internet censored countries, based of the the most recent Freedom House report [8], which resulted in the countries displayed in 4.4. The HTTPS scan shows a first broad classification of the scan, in which countries like China and Iran showed strong indicators for possible censorship events happening nationally, while countries like

Myanmar and Russia showed a surprisingly low amount of indicators. When analyzing 4.5 it is interesting to note that despite the fact that the request was sent in plain text using only HTTP, there is a decrease in anomalies witnessed.

While one would suggest that censoring HTTP requests would take less technological effort and therefore increase the possible vectors for censorship, like seen in the increase in Vietnam, the worldwide steep adaptation of TLS within the Internet, could force countries like China tend to rather focus in on analyzing HTTPS than HTTP.

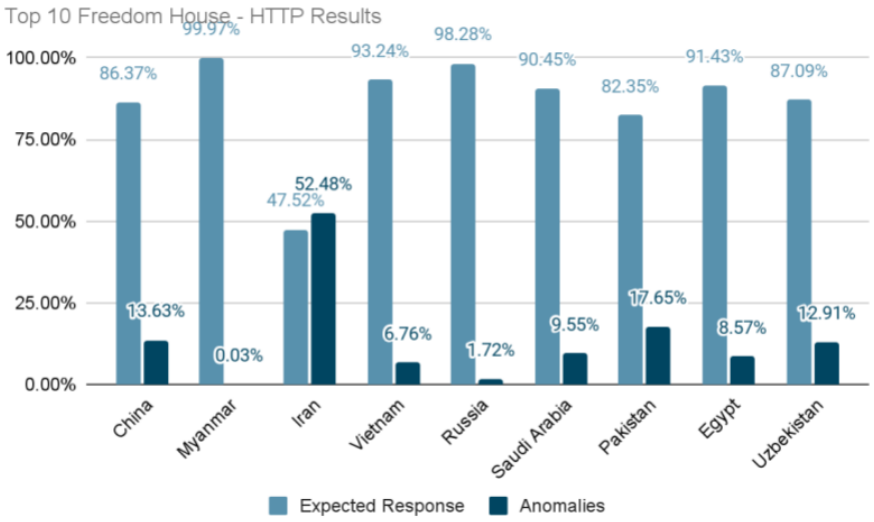


Figure 4.5: Overview of matching and mismatching responses within the HTTP result set.

However, researcher measuring Internet censorship are faced with several key challenges as outlined by Raman *et al.* [7], one of which being, that there is never just one single strong and clear indicator for censorship, but rather several tender ones that need to be accumulated in order to get a holistic view on an possible censorship event.

So while countries like China and Iran already show strong indicators for ongoing censorship events, the lack there of in countries like Russia might be due to the fact that a closer inspection is needed to find the truth. Figure 4.6 gives an overview of the location based metrics of the infrastructural web servers collected prior to the scan.

While stats based on the population size show that the selection of the web servers in countries like China and Vietnam seem to be sufficient, it is lacking for the case of Russia. The APNIC data states that there are currently 108 AS networks registered, however the collected data shows that it only covers 11. While

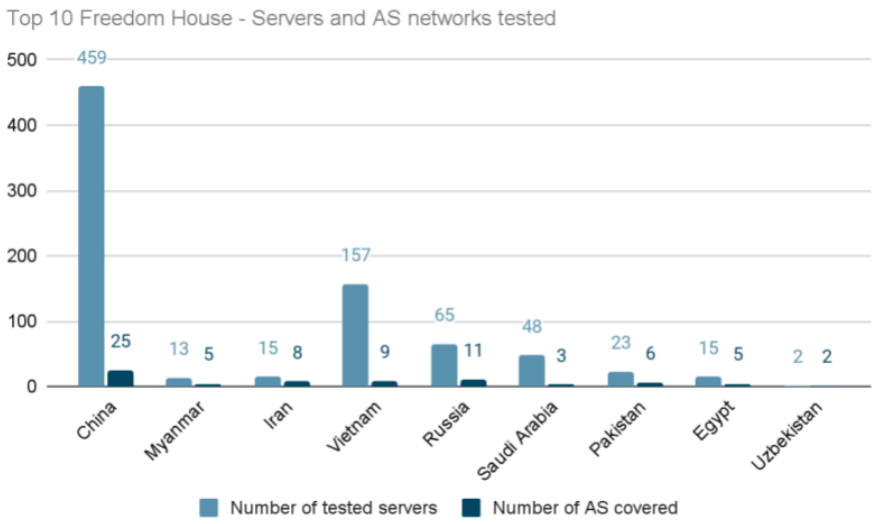


Figure 4.6: Country and location information of the probed infrastructural web servers.

observatories like OONI [3] are dependent on volunteers running local probes within a country, remote measurements techniques, like the one used for this thesis, are not restricted to individuals but are clearly have their limits on how many of the actual censorship events they can observe[52].

The next step within analyzing the results focuses on actual error responses from the anomalies received, which can be categorized into the following classes:

- Blockpages - The response received matches an publicly [63] know blockpage. This category serves as one of the clearest and strongest indicator for censorship. If a blockpage has been received, a censorship device must have been located on the path between the measurement machine or directly within the ISP. The blockpage was then either directly injected and served as a response or the probe was redirected to another machine, which serves the blockpage.
- Body anomaly - The response body does not matched the template, which indicates that certain keywords triggered a different response then from the control probe.
- Empty HTTP - The generated control template expected content from the HTTP body, but no content was served.
- Header anomaly - The HTTP headers mismatch the template.
- Invalid HTTP - The Python script was not able to properly parse the HTTP response, due to content-length mismatches or invalid encoding.

[3] https://ooni.org/

Anomaly Category	Total Occurrence	Relative Occurrence
Blockpages	511	0.0018%
Body anomaly	983	0.0034%
Empty HTTP	1273	0.0045%
Header anomaly	1008	0.3530%
Invalid HTTP	33	0.0001%
Status anomaly	14088324	49.3410%
TCP failed	334721	1.1723%
TCP reset	14077141	49.3019%
TCP timeout	42005	0.1471%
TLS anomaly	3459	0.0121%
TLS failed	3479	0.0122%

Table 4.1: Comparing the causes for anomalies

- Status anomaly - The most common example for this within the data set are expected 200 codes, but 403 codes were served. Most likely due to a censorship device that has been triggered.
- TCP failed - The TCP connection failed due to a protocol error or was refused by the server.
- TCP reset - With the blockpage category the most common and strongest indicator for a censorship event. Most likely a RST-packet was injected on the path.
- TCP timeout - A strong indicator for a censorship device simply dropping the probe request before it reaches the server.
- TLS anomaly - Includes mismatches within the server certificate, the cipher suites or the TLS version served.
- TLS failed - The TLS handshake wasn't established, usually due to a protocol error.

The data displayed in 4.1 gives an overview anomaly causes within the HTTPS scan results. With the two most prevalent causes being anomalies within the header fields and RST-packets being injected into the connection.

In order to determine which keywords within an HTTPS request are most susceptible for triggering a censorship device, all anomalies have been mapped to their domain category classes and 4.2 reflects what percentage of domains are being affected within the each category classes.

On a global scale, topics surrounding pornography, social networking, media sharing, news media and

Domain Category	World	China	Myanmar	Iran	Vietnam	Russia
Alcohol & Drugs	0.69%	1.15%	0.28%	66.82%	1.93%	0.00%
Anonymization & circumvention tools	2.74%	15.36%	0.47%	85.14%	0.86%	0.27%
Communication Tools	2.83%	31.58%	0.14%	51.46%	0.72%	0.15%
Culture	3.96%	55.40%	0.05%	59.75%	0.42%	0.38%
E-commerce	1.72%	22.41%	0.05%	30.03%	0.49%	0.24%
Economics	0.52%	0.88%	0.06%	15.25%	0.18%	0.49%
File-sharing	2.50%	28.57%	0.95%	35.64%	1.03%	0.00%
Gambling	2.60%	11.31%	0.82%	76.19%	5.47%	0.00%
Gaming	1.21%	2.09%	0.08%	43.80%	0.25%	0.56%
Hacking Tools	1.08%	1.48%	0.43%	47.03%	2.78%	0.15%
Hate Speech	0.66%	0.77%	0.00%	50.00%	0.00%	0.00%
History arts and literature	0.34%	1.19%	0.18%	75.00%	0.00%	0.00%
Human Rights Issues	1.98%	34.06%	0.10%	49.32%	0.07%	0.54%
LGBT	1.02%	11.92%	0.11%	64.51%	0.37%	0.14%
Media Sharing	6.03%	62.11%	0.52%	83.08%	1.11%	1.33%
Miscellaneous content	0.37%	0.55%	0.33%	60.00%	1.71%	0.00%
News Media	4.31%	63.41%	0.15%	63.65%	0.09%	0.73%
Online Dating	1.59%	1.69%	0.13%	88.13%	0.81%	0.00%
Political Criticism	3.63%	54.41%	0.06%	40.00%	0.04%	0.00%
Pornography	6.82%	44.38%	0.98%	95.60%	7.18%	4.45%
Provocative Attire	1.05%	1.02%	0.21%	97.83%	0.74%	0.78%
Public Health	0.92%	6.57%	0.09%	22.12%	0.25%	0.00%
Religion	0.67%	10.65%	0.11%	58.78%	0.00%	0.00%
Search Engines	1.96%	25.12%	0.10%	33.80%	0.17%	0.00%
Sex Education	1.35%	17.86%	0.28%	55.90%	0.19%	0.00%
Social Networking	5.97%	61.91%	0.65%	79.25%	1.50%	0.93%

Table 4.2: Overview of the percentage of domains blocked within their respected categories.

culture seem to be among the most effected categories. Depending on the actual location of the web server, the trend can shift dramatically, like websites about political criticism and human rights issues in China or websites about censorship circumvention, alcohol, gambling and online dating in Iran.
If the location of the web server would not be take into account, then keywords surrounding themes like economics, public health and public health seem to be among the safest choices.

4.2 Creating Synthetic Content

The following section presents an experiment in which a website is created using GPT-3. The experiment utilizes the OpenAI Python library to generate the HTML structure, text, and image content for the website. The goal of the experiment is to showcase the capabilities of GPT-3 in terms of website creation, including its ability to generate coherent text content and its ability to understand and follow website structure.

4.2.1 Methodology

Based on the results in section 4.1.3 it was decided on the approach to generate a blog like website, that would be able to reflect various kinds of content and therefore apply to the broad need of offering various forms of synthetic content based on the country the probe-resistant proxy resides in.

As a first step an account needs to be registered at OpenAI's website [4] in order to be able to generate an API key which can be used to send requests to GPT-3 pragmatically.

The library offers several modules, but only the following will be tested and developed on for this thesis:

- openai.Completion - Allows for general text generation.
- openai.Code - Interface for generating code based on a descriptive task.
- openai.Image - Provides access to generate images based on text input.

During the development and testing of the Python application both the completion as well as the code module were used to generate an HTML template that reflects the DOM structure of a real word blog website. This involved extensive testing of numerous prompts in order to precisely generate a template that can be further parsed with, without adjusting the parameters to return a deterministic response for every request and therefore save the ability to create random websites for each execution.

Once the ideal parameters have been determined, a parsing algorithm was developed using the Beautiful Soup library [5], which parses the returned DOM structure and checks for certain HTML tags that will need to be populated with content.

The application asks the user for command line input on what the general theme of the website should be and how many blog entries should be created. Based on that information the content for the first blog post will be generated using GPT-3 and populated within the correct DOM node.

After the blog content has been generated, an image will be created using the image module, which will be embedded inside the content to provide the blog with more authenticity. Prior to generating the image, a

[4] https://openai.com/api/

[5] https://www.crummy.com/software/BeautifulSoup/bs4/doc/

request will be issued to describe a prompt for the generated blog and provide this information to the image prompt for generating a suitable image.

4.2.2 Testing Prompts

To generate the ideal prompt for creating the structure of the website and the accompanying parameter options, a total of 320 variations have been tested, for 3 different engines, resulting in a equal amount of HTML templates that were compared to each other to find the most authentic and best suitable option.

```
code = openai.Completion.create(
    engine="text-davinci-002",
    prompt=prompt,
    max_tokens=4000,
    best_of=2,
    temperature=1
).choices[0].text
```

Listing 4.4: Code snippet for generating text through the OpenAI API

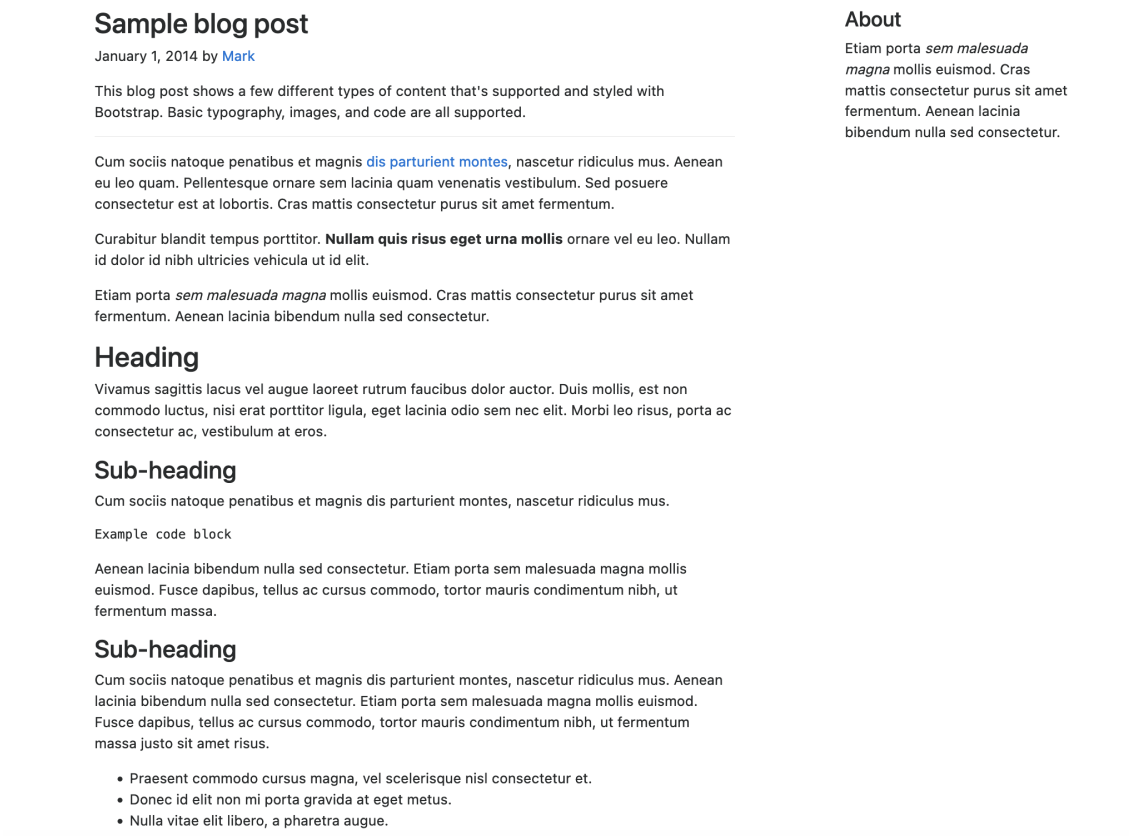
4.4 displays the option parameters tested for this experiment. The `engine` option allows to choose which model to use for the task. For this experiment the models text-davinci-003, text-davinci-002 and code-davinci-002 were chosen. Each of the tested models allowed a maximum of 4097 tokens for the model to generate the output, however this includes the amount of input tokens the prompt would generate. The `temperature` parameter allows to adjust the level of creativity of the generated output, with the default level being 0.5. A low level would result in a more deterministic behavior, generating outputs that are more repetitive. With the `best_of` options multiple results for one requested prompt can be generated, to afterwards be compared against each other.

Early on it was clear that the code-davinci-002 model was not fitted for the purpose of this experiment. While this model was developed for the particular use case of generating even complex code using a descriptive text prompt, it is only able to handle the creation of code for one particular language. Therefore it is not able to generate an HTML structure with css embedding within it for example.

And while the text-davinci-003 models is stated to be a more capable model than its predecessor text-davinci-002, it was the latter model that was able to produce more authentic templates. After manually comparing all of the 320 generated websites, the following prompt was finalized for the task of generating the HTML template:

> Create a skeleton that reflects a personal blog website with bootstrap and a responsive design that is optimized for mobile as well. Avoid using navigation bars, all the blog posts should be listed on the main page. Add dummy links to the blog entries. Import all the necessary scripts before the closing body tag, such as jquery, popper and bootstrap.

The figure 4.8 displays one of the sample websites returned by the prompt and will be used for the rest of this result section as before and after example. In the next step Python application parses the HTML structure.



Sample blog post

January 1, 2014 by Mark

This blog post shows a few different types of content that's supported and styled with Bootstrap. Basic typography, images, and code are all supported.

Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aenean eu leo quam. Pellentesque ornare sem lacinia quam venenatis vestibulum. Sed posuere consectetur est at lobortis. Cras mattis consectetur purus sit amet fermentum.

Curabitur blandit tempus porttitor. **Nullam quis risus eget urna mollis** ornare vel eu leo. Nullam id dolor id nibh ultricies vehicula ut id elit.

Etiam porta *sem malesuada magna* mollis euismod. Cras mattis consectetur purus sit amet fermentum. Aenean lacinia bibendum nulla sed consectetur.

Heading

Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros.

Sub-heading

Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

```
Example code block
```

Aenean lacinia bibendum nulla sed consectetur. Etiam porta sem malesuada magna mollis euismod. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa.

Sub-heading

Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aenean lacinia bibendum nulla sed consectetur. Etiam porta sem malesuada magna mollis euismod. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus.

- Praesent commodo cursus magna, vel scelerisque nisl consectetur et.
- Donec id elit non mi porta gravida at eget metus.
- Nulla vitae elit libero, a pharetra augue.

About

Etiam porta *sem malesuada magna* mollis euismod. Cras mattis consectetur purus sit amet fermentum. Aenean lacinia bibendum nulla sed consectetur.

Figure 4.7: Response generated by GPT-3 using the prompt in 4.2.2.

After the template has been successfully parsed, the application starts to populate the DOM nodes that have been identified as tags that need additional content, like `<h1>` and `<p>`. At this point the application issues the second GPT-3 request, asking the user through the command line prompt what the general topic or theme of the website should be. In this case the input `social economics` was provided, resulting in the following prompt:

> Create a blog post for a website that covers the basic topics of social economics and add HTML tags to it.

The application takes the output and inserts the content into the placeholder, identified prior by the parser, for the first blog post. In order to increase the authenticity of the blog post, a image will be generated and inserted into the DOM. To do so, the previously generated blog content will be removed from all HTML tags and inserted into a new prompt that will be used to create an appropriate image for it:

> Describe a image that fits this text: + `blog`

The outcome is a short sentence that GPT determines describes the blog content appropriately. All that is left to do is to use the image module to request the image generation. The function allows for 3 different sizing options, the Python application uses by default the highest resolution 4.5.

```
image = openai.Image.create(
    prompt=keyphrase,
    n=1,
    size="1024x1024",
    response_format="b64_json",
)
```

Listing 4.5: Code snippet for generating an image through the OpenAI API

By default, GPT would respond with an URL containing the generated image. However, it provides the option to receive the image with a Base64-encoding and save it locally. In the final step the application inserts the image within the blog content.

The application provides the option to repeat these steps several times, to create multiple blog entries, where the previous entries are used to instruct GPT to avoid generating similar text.

4.2.3 Results

Prompt	Token	Price	Time
Creating the website template	1320	0.1584$	41.29s
Creating a blog post	321	0.0039$	15.121s
Describe an image based on blog	329	0.0039$	1.99s
Create an image based on input	-	0.02$	6.86s
Additional content filling (e.g. about section)	278	0.0453$	21.16s

Table 4.3: GPT-3 prompt benchmarks.

Table 4.3 provides an overview of the average amount of tokens used and an average cost metric based on it, as well as the average execution time for each prompt, resulting in a total cost of about 0.21$ for the generation of a blog website that contains one blog entry. Each follow up blog will result in additional costs of 0,0026$. Unsurprisingly, the prompt which used up the most tokens, creating the website template, is the one with slowest execution time.

Figure 4.8 displays a screenshot of the finalized blog.

Figure 4.8: Final website generated by the Python application.

5 Conclusion

A key focus of this thesis has been on the use of GPT-3, a powerful natural language processing technology, to generate authentic-looking websites that can be used to disguise proxy servers as harmless web servers to censoring authorities. Through a series of experiments, the paper has revealed which keyword categories are most susceptible to censorship, and has demonstrated the effectiveness of using GPT-3 to generate text and image content for these websites. The results of this research suggest that GPT-3 and other advanced AI technologies have the potential to play a significant role in the fight against internet censorship, by providing powerful tools for those working to promote freedom of expression and access to information online.

5.1 Future Work

There are several areas that can be improved upon the Python application developed for this thesis. One of the key areas is to provide more configuration options for the user, allowing them to customize the website to their specific needs. Additionally, supporting a wider range of website styles could also be beneficial. Optimizing the prompts used to generate the text and image content could also lead to better results. Another important aspect is to work towards lower costs and increased efficiency, this could be achieved through better error handling and by finding ways to reduce the number of API calls required. It would be interesting to see the application evolve to support even more complex websites, such as those that contain multiple linked pages and a navigation bar. This could be done by developing the application to support website structure generation, rather than just content generation.

List of Figures

List of Tables

Listings

Acronyms

AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
ANN	Artificial Neural Networks
API	Application Programming Interface
AS	Autonomous System
DNS	Domain Name System
DOM	Document Object Model
DPI	Deep Packet Inspection
Great Firewall of China	GFW, is the combination of legislative actions and technologies enforced by the People's Republic of China to regulate the Internet domestically.
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IP	Internet Protocol
ISP	Internet Service Provider
IXP	Internet Exchange Point
JSON	JavaScript Object Notation
NLP	Natural Language Processing

PT	Pluggable Transports
RNN	Recurrent Neural Networks
Server Name Indication	SNI, is an extension to the TLS protocol by which a client indicates which hostname it is attempting to connect to at the start of the handshaking process.
SSH	Secure Shell Protocol
TCP	Transmission Control Protocol
TCP Reset packet	RST-packet, a TCP packet with the RST bit set within the TCP header to terminate the connection between a client and server
TLS	Transport Layer Security
Tor	The Onion Router
VPN	Virtual Private Network

Bibliography

[1] Barney Warf, *Global geographies of the internet*. Springer Science & Business Media, 2012.

[2] Jack L Goldsmith, "Against cyberanarchy," *The University of Chicago Law Review*, vol. 65, no. 4, pp. 1199–1250, 1998.

[3] Neal Kumar Katyal, "Criminal law in cyberspace," *University of Pennsylvania Law Review*, vol. 149, no. 4, pp. 1003–1114, 2001.

[4] Nart Villeneuve, "The filtering matrix: Integrated mechanisms of information control and the demarcation of borders in cyberspace," *First Monday*, 2006.

[5] Johan Eriksson and Giampiero Giacomello, "Who controls what, and under what conditions?" *International Studies Review*, vol. 11, no. 1, pp. 206–210, 2009.

[6] Jessica E Bauml, "It's a mad, mad internet: Globalization and the challenges presented by internet censorship," *Fed. Comm. LJ*, vol. 63, p. 697, 2010.

[7] Ram Sundara Raman, Adrian Stoll, Jakub Dalek, Reethika Ramesh, Will Scott, and Roya Ensafi, "Measuring the deployment of network censorship filters at global scale.," in *NDSS*, 2020.

[8] *Freedom on the net 2022 - countering an authoritarian overhaul of the internet*, https://freedomhouse.org/report/freedom-net/2022/countering-authoritarian-overhaul-internet, Online; Accessed: 2023-01-20.

[9] Diwen Xue, Benjamin Mixon-Baca, Anna Ablove, Beau Kujath, Jedidiah R Crandall, and Roya Ensafi, "Tspu: Russia's decentralized censorship system," in *Proceedings of the 22nd ACM Internet Measurement Conference*, 2022, pp. 179–194.

[10] Seth F Kreimer, "Technologies of protest: Insurgent social movements and the first amendment in the era of the internet," *University of Pennsylvania Law Review*, vol. 150, no. 1, pp. 119–171, 2001.

[11] Barney Warf and John Grimes, "Counterhegemonic discourses and the internet," *Geographical Review*, vol. 87, no. 2, pp. 259–274, 1997.

[12] Paul Pearce, Ben Jones, Frank Li, Roya Ensafi, Nick Feamster, Nick Weaver, and Vern Paxson, "Global measurement of {dns} manipulation," in *26th USENIX Security Symposium (USENIX Security 17)*, 2017, pp. 307–323.

[13] Peter C Johnson, Apu Kapadia, Patrick P Tsang, and Sean W Smith, "Nymble: Anonymous ip-address blocking," in *International Workshop on Privacy Enhancing Technologies*, Springer, 2007, pp. 113–133.

[14] Nick Feamster, Magdalena Balazinska, Greg Harfst, Hari Balakrishnan, and David Karger, "Infranet: Circumventing web censorship and surveillance," in *11th USENIX Security Symposium (USENIX Security 02)*, 2002.

[15] *Technical multi-stakeholder report on internet shutdowns: The case of iran amid autumn 2022 protests*, https://ooni.org/post/2022-iran-technical-multistakeholder-report/, Online; Accessed: 2023-01-20.

[16] Diwen Xue, Reethika Ramesh, Leonid Evdokimov, Andrey Viktorov, Arham Jain, Eric Wustrow, Simone Basso, and Roya Ensafi, "Throttling twitter: An emerging censorship technique in russia," in *Proceedings of the 21st ACM Internet Measurement Conference*, 2021, pp. 435–443.

[17] Patricia Vargas-Leon, "Tracking internet shutdown practices: Democracies and hybrid regimes," in *The turn to infrastructure in Internet governance*, Springer, 2016, pp. 167–188.

[18] Roya Ensafi, Philipp Winter, Abdullah Mueen, and Jedidiah R Crandall, "Analyzing the great firewall of china over space and time.," *Proc. Priv. Enhancing Technol.*, vol. 2015, no. 1, pp. 61–76, 2015.

[19] Philipp Winter and Stefan Lindskog, "How china is blocking tor," *arXiv preprint arXiv:1204.0447*, 2012.

[20] Roger Dingledine, Nick Mathewson, and Paul Syverson, "Tor: The second-generation onion router," Naval Research Lab Washington DC, Tech. Rep., 2004.

[21] Arun Dunna, Ciarán O'Brien, and Phillipa Gill, "Analyzing china's blocking of unpublished tor bridges," in *8th USENIX Workshop on Free and Open Communications on the Internet (FOCI 18)*, 2018.

[22] *New bridgedb version using rdsys*, https://forum.torproject.net/t/tor-project-new-bridgedb-version-using-rdsys/2335, Online; Accessed: 2023-01-20.

[23] Philipp Winter, Tobias Pulls, and Juergen Fuss, "Scramblesuit: A polymorphic network protocol to circumvent censorship," in *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*, 2013, pp. 213–224.

[24] *Tor's circumvention manual*, https://tb-manual.torproject.org/circumvention/, Online; Accessed: 2023-01-20.

[25] *Obfs4*, https://gitlab.torproject.org/tpo/anti-censorship/pluggable-transports/obfs4, Online; Accessed: 2023-01-20.

[26] *Snowflake*, https://snowflake.torproject.org/, Online; Accessed: 2023-01-20.

[27] Zhongjiang Yao, Jingguo Ge, Yulei Wu, Xiaodan Zhang, Qiang Li, Lei Zhang, and Zhuang Zou, "Meek-based tor traffic identification with hidden markov model," in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, IEEE, 2018, pp. 335–340.

[28] Jan Beznazwy and Amir Houmansadr, "How china detects and blocks shadowsocks," in *Proceedings of the ACM Internet Measurement Conference*, 2020, pp. 111–124.

[29] Paul Ferguson and Geoff Huston, "What is a vpn?," 1998.

[30] Thomas Berger, "Analysis of current vpn technologies," in *First International Conference on Availability, Reliability and Security (ARES'06)*, IEEE, 2006, 8–pp.

[31] Oliver Farnan, Joss Wright, and Alexander Darer, "Analysing censorship circumvention with vpns via dns cache snooping," in *2019 IEEE Security and Privacy Workshops (SPW)*, IEEE, 2019, pp. 205–211.

[32] Sergey Frolov, Jack Wampler, and Eric Wustrow, "Detecting probe-resistant proxies.," in *NDSS*, 2020.

[33] Sergey Frolov and Eric Wustrow, "{Httpt}: A {probe-resistant} proxy," in *10th USENIX Workshop on Free and Open Communications on the Internet (FOCI 20)*, 2020.

[34] Zhi-Hua Zhou, *Machine learning*. Springer Nature, 2021.

[35] Issam El Naqa and Martin J Murphy, "What is machine learning?" In *machine learning in radiation oncology*, Springer, 2015, pp. 3–11.

[36] Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman, "Natural language processing: An introduction," *Journal of the American Medical Informatics Association*, vol. 18, no. 5, pp. 544–551, 2011.

[37] Nitin Indurkhya and Fred J Damerau, *Handbook of natural language processing*. Chapman and Hall/CRC, 2010.

[38] Steven Walczak, "Artificial neural networks," in *Advanced methodologies and technologies in artificial intelligence, computer simulation, and human-computer interaction*, IGI global, 2019, pp. 40–53.

[39] *What is a neural network*, https://www.tibco.com/reference-center/what-is-a-neural-network, Online; Accessed: 2023-01-20.

[40] Anders Krogh, "What are artificial neural networks?" *Nature biotechnology*, vol. 26, no. 2, pp. 195–197, 2008.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[42] Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao, "Learning deep transformer models for machine translation," *arXiv preprint arXiv:1906.01787*, 2019.

[43] Ilya Sutskever, Oriol Vinyals, and Quoc V Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[44] *The illustrated transformer*, https://jalammar.github.io/illustrated-transformer/, Online; Accessed: 2023-01-20.

[45] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, *et al.*, "Improving language understanding by generative pre-training," 2018.

[46] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[47] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[48] Luciano Floridi and Massimo Chiriatti, "Gpt-3: Its nature, scope, limits, and consequences," *Minds and Machines*, vol. 30, no. 4, pp. 681–694, 2020.

[49] Kevin Bock, George Hughey, Xiao Qiang, and Dave Levin, "Geneva: Evolving censorship evasion strategies," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 2199–2214.

[50] Gabriel Kaptchuk, Tushar M Jois, Matthew Green, and Aviel D Rubin, "Meteor: Cryptographically secure steganography for realistic distributions," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 1529–1548.

[51] Steven Sheffey and Ferrol Aderholdt, "Improving meek with adversarial techniques," in *9th USENIX Workshop on Free and Open Communications on the Internet (FOCI 19)*, 2019.

[52] Benjamin VanderSloot, Allison McDonald, Will Scott, J Alex Halderman, and Roya Ensafi, "Quack: Scalable remote measurement of {application-layer} censorship," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018, pp. 187–202.

[53] Fatemah Alharbi, Michalis Faloutsos, and Nael Abu-Ghazaleh, "Opening digital borders cautiously yet decisively: Digital filtering in saudi arabia," in *10th USENIX Workshop on Free and Open Communications on the Internet (FOCI 20)*, 2020.

[54] Ram Sundara Raman, Prerana Shenoy, Katharina Kohls, and Roya Ensafi, "Censored planet: An internet-wide, longitudinal censorship observatory," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 49–66.

[55] Zakir Durumeric, Eric Wustrow, and J Alex Halderman, "{Zmap}: Fast internet-wide scanning and its security applications," in *22nd USENIX Security Symposium (USENIX Security 13)*, 2013, pp. 605–620.

[56] *Apnic is the regional internet registry administering ip addresses for the asia pacific*, https://www.apnic.net/, Online; Accessed: 2023-01-20.

[57] *Center for applied internet data analysis based at the university of california's san diego supercomputer center*, https://www.caida.org/, Online; Accessed: 2023-01-20.

[58] *Maxmind provides ip intelligence through the geoip brand.* https://www.maxmind.com/, Online; Accessed: 2023-01-20.

[59] *Peeringdb is a freely available, user-maintained, database of networks, and the go-to location for interconnection data.* https://www.peeringdb.com, Online; Accessed: 2023-01-20.

[60] *Censys is a web-based search platform for assessing attack surface for internet connected devices.* https://censys.io/, Online; Accessed: 2023-01-20.

[61] *Contained are url testing lists intended to help in testing url censorship, divided by country codes.* https://github.com/citizenlab/test-lists, Online; Accessed: 2023-01-20.

[62] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen, "Tranco: A research-oriented top sites ranking hardened against manipulation," in *Proceedings of the 26th Annual Network and Distributed System Security Symposium*, ser. NDSS 2019, Feb. 2019. DOI: 10.14722/ndss.2019.23386.

[63] *Censored planet's blockpage signature database*, https://assets.censoredplanet.org/blockpage_signatures.json, Online; Accessed: 2023-01-20.