The Great Firewall's active probing circumvention technique with port knocking and SDN

Pavel Liubinskii

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 31.12.2020

Supervisor

Prof. Jukka Manner

Advisor

MSc Markus Peuhkuri

Aalto University
School of Electrical Engineering

Copyright © 2020 Pavel Liubinskii

Aalto University
School of Electrical Engineering

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author	Pavel Liubinskii				
Title	The Great Firewall's active probing circumvention technique with port knocking and SDN				
Degree programme	Computer, Communication and Information Sciences				
Major	Communications Engineering			**Code of major**	ELEC24
Supervisor	Prof. Jukka Manner				
Advisor	MSc Markus Peuhkuri				
Date	31.12.2020	**Number of pages**	65	**Language**	English

Abstract

This thesis observes the phenomenon of online censorship, both from blocking and mitigation perspectives. It enumerates and characterizes typical methods and types of Internet censorship, as well as effective circumvention solutions and strategies. Additionally, the study provides detailed observations of the Great Firewall, the ultimate weapon of a Chinese censor, and the Tor anonymity network, the broadly recognized anti-censorship and anti-surveillance tool. Furthermore, it illuminates the Tor network blocking and the firewall's scanning engine (active probing), which is used to detect mitigating servers. The results of the study indicate that 1) The Tor network is heavily suppressed in China; 2) Active probing technique still contributes to blocking decisions; and 3) The Great Firewall successfully engages blocking against obfs4 Tor bridges. Finally, the work suggests a solution for bypassing the Great Firewall using a traffic engineering approach, i.e., software-defined networking and the well-known port knocking technique.

Keywords Censorship circumvention, The Great Firewall, Active probing, Tor, Tor bridges

Contents

Abbreviations

AEAD	Authenticated Encryption with Associated Data
AS	Autonomous Systems
BGP	Border Gateway Protocol
CA	Certificate Authority
CDN	Content Delivery Network
DNS	Domain Name Systems
DNSSEC	Domain Name System Security Extensions
DoH	DNS Queries over HTTPS
DoT	DNS over TLS
DPI	Deep Packet Inspection
ESNI	Encrypted Server Name Indication
GFW	Great Firewall of China
IANA	Internet Assigned Numbers Authority
ISP	Internet Service Provider
MiTM	Man In The Middle
NIDS	Network Intrusion Detection System
NIPS	Network Intrusion Prevention System
OP	Onion proxy
OR	Onion routers
OTP	One-Time Password
PT	Pluggable Transport
QoS	Quality of Service
SDN	Software-defined Networks
SNI	Server Name Indication
TCIS	Tor Connection Initiation Simulator
TLS	Transport Layer Security
VoIP	Voice over Internet Protocol
VPN	Virtual Private Network

1 Introduction

Over the past decade, internet censorship circumvention has been the subject of growing interest due to pervasive restriction practices used by numerous states. It is well known that China is the country with the most severe internet censorship. Chinese internet restrictions started with the development of the Great Firewall (GFW) in 1996 [1]. Nowadays, the firewall restricts access to many popular internet services, including Facebook and Google. The GFW is known to be the most sophisticated network censorship solution in the world. However, the GFW is not perfect, which makes it possible to bypass the restrictions.

In recent years, many efforts have been devoted to investigating GFW operations and developing countermeasures. It was shown that the firewall relies on almost all known network filtering techniques [1]. Moreover, the usage of reactive network scanning, known as active probing, has been detected and shown to be used by GFW against Tor and other anti-censorship tools[2, 3, 4]. It was recently reported that the firewall had started complete blocking of some encrypted connections regardless of their purpose [5]. Nevertheless, numerous anti-censorship solutions and protocols have been developed by both the academic community and business. Some notable examples include domain fronting[6], which is actively used by Tor; MTProto protocol from Telegram [7]; Shadowsocks[8]; and Lampshade [9]. Unfortunately, the continuous censorship arms race increasingly challenges current solutions. Recent advances in machine learning have allowed detecting Shadowsocks traffic with about 90% probability[10]. Researchers from the University of Colorado have demonstrated the detection of five popular anti-censorship solutions by observing their network behavior [11].

Existing anti-censorship tools make the Chinese censorship strategy ineffective by relying on possible high collateral damage connected with blocking or by obfuscating and hiding communications using an out-of-band delivered secret. The first strategy is implemented by domain fronting and is used in Tor meek transport, while the second strategy is inherited by Shadowsocks, obfs4, and many other anti-censorship tools. However, domain fronting is expensive to operate since it relies on third-party infrastructure. Collateral damage might be acceptable to the censor, as occurred during the recent Belarussian Internet shutdown. The problem with dedicated anti-censorship protocols lies in their exclusive circumvention purpose. Usage of pervasive obfuscation and encryption differentiates their traffic patterns from general-purpose protocols, and makes them suspicious in the eyes of a censor. This significantly facilitates circumvention detection [11].

The GFW relies on active probing, a specially constructed forge connection request, to identify circumvention servers. Numerous anti-censorship solutions avoid active probing by using an out-of-band delivered secret in order to encrypt connections. However, these solutions all operate on an application level and do not attempt to identify probing on a lower network level.

A promising strategy to identify GFW probes would be to send a specially constructed packet (i.e., a knock packet) from a censored PC to a circumvention infrastructure, thus facilitating censorship bypass for several traditional protocols.

This packet reliably identifies the legitimate client address, whereas all other addresses should be considered as probe sources and treated accordingly. The suggested strategy would ensure better censorship penetration for a broader number of tunneling and proxy protocols.

This study aims to demonstrate how the GFW blocks Tor, observe GFW's scanning (probing) activities, design a method for identifying and blocking the observed GFW probes, and evaluate the method's effectiveness in terms of bypassing censorship. To achieve these objectives, the tor bridge's circumvention performance will be evaluated and compared before and after the proposed method's deployment. This work will only focus on active probing. Other filtering techniques are excluded from the scope due to their well-defined effectiveness and well-known bypassing methods.

The rest of this thesis is divided into seven chapters. Chapter 2 reviews online censorship in general. The Great Firewall and the Tor anonymity network are discussed in Chapters 3 and 4. Chapter 5 demonstrates the Tor network reachability from mainland China. Chapter 6 presents an examination of the GFW probing activities observed in this work. Chapter 7 introduces a method for identifying and blocking the GFW probes and demonstrates the effectiveness of the suggested method. A conclusion, possibilities for future studies, and some limitations are discussed in Chapter 8.

2 Internet censorship

The Internet has fundamentally changed every aspect of people's lives and has become a modern world engine. In 2020, the top five world biggest companies by market value are Microsoft, Apple, Amazon, Alphabet (Google), and Facebook[12]. This list radically illustrates the importance of the global network to our economy and society. This transformation happened because of the content and services that are available on the Internet. Eventually, free borderless content distribution became the foundation of the welfare of the entire world.

However, the Internet contains some content that regulators, lawmakers, and officials would like to block. The content might be restricted based on the user's age or location using various online censorship methods. The incentives for censorship are diverse and include preventing infringement of intellectual property rights, stopping the spread of child abuse materials, countering illegal activities on the Internet, and protecting national security[13].

When talking about Internet censorship, the words blocking, filtering, and censorship and equally treated regardless of their connotation. Although the word "censorship" has a strong negative meaning, whereas "filtering" sounds more innocent and harmless. This study observes internet censorship from a technical perspective and discusses blocking and filtering of online content regardless of censors' motives and reasons. However, some types of censorship are presented in the upcoming section, with the goal to provide a broader view of the studied question.

2.1 Types of online censorship

Public authorities restrict access to information and related services that are illegal in a particular jurisdiction, considered a threat to public welfare, or unacceptable to any audience. Furthermore, private and public companies imply blocking based on their network security policies and codes of conduct. Copyright holders use legislative instruments to remove piracy content from a search engine output and hosting services. Additionally, political and ideological censorship represents a different type of censorship because of its significance. Although this type of censorship might be merged with illegal content blocking if not to consider its political nature. Nevertheless, three types of censorship are enumerated; these include:

– Harmful and illegal content blocking

– Business censorship

– Political and ideological censorship

2.1.1 Harmful and Illegal content blocking

The specifics of the local legal environment have an impact on content blocking. Content blocking usually occurs if it violates intellectual property laws, considered a threat to national security, or banned for cultural or political reasons[13]. One of the difficulties, which force the authorities of individual countries to utilize censorship measures, comes from the differences in the legislative definition of illegal (harmful) content by different states.

For example, European digital single market legislation defines categories of illegal and harmful content, such as terrorist content, xenophobic or racist speech, child sexual abuse material, infringements of intellectual property rights, commercial scams, and frauds[14]. Another example demonstrates the definition of harmful content from the United Kingdom parliament. This list is more detailed than European classification and, in addition to the above categories, adds such items as immigration crime, modern slavery, promotion of Female Genital Mutilation (FGW), encouraging or assisting suicide, and disinformation[15]. United States legislation, such as the Stop Advertising Victims of Exploitation Act (SAVE) [16] and Children's Internet Protection Act (CIPA)[17], defines almost similar categories of harmful and illegal data as European and UK documents. In addition to some previously mentioned content categories, Russia labels information related to narcotics, suicides, and the promotion of homosexual relations as harmful and illegal[18].

These examples not only show that the meaning of Illegal content varies from country to country but also separate categories that are globally recognized by most countries as harmful and illegal, including three universal and significant categories:

– Terrorist content

– Child sexual abuse material

– Infringements of intellectual property rights

These three categories are further observed from the European legislative process's position and based on the European Commission's documents. For instance, Article 21 of the EU directive 2017/541 of the European Parliament and of the Council of 15 March 2017 on combating terrorism [19] states that EU Member States shall take the necessary measures, such as content removal or blocking, to prevent the distribution of online materials with the intent or a provocation of a terrorist offense.

Europe's directive 2011/92/EU on combating the sexual abuse and sexual exploitation of children and child pornography [20] aims to protect children from sexual abuse behavior and deny access to child abuse materials. Article 21 of the directive empowers Member States to take measures against the dissemination of material related by the directive as child abuse content. Furthermore, the directive establishes grounds to prosecute owners and distributors of child abuse materials, and censorship acts as a tool in this process with the goal to protect children.

Controversial EU directive 2019/790 on copyright and related rights in the Digital Single Market [21] forces online content-sharing service to remove unauthorized materials and establishes liability for them if content platforms fail to remove unauthorized materials and demonstrate that they have done their best to prevent the future uploading of specific unauthorized works. Thus, content providers in Europe are obligated to establish self-censorship to prevent the uploading of unauthorized content. The directive raised a hot discussion both in the European parliament and society. On 26 March 2019, the directive was approved by 349 deputies. Two hundred seventy-four parliament members voted against the directive. After the vote, Finland, Italy, Luxemburg, the Netherlands, and Polland issued a joint statement that noted that the directive fails to maintain the right balance between copyright holders' protection and the interest of society and business[22].

The last example demonstrates how controversial and ambiguous can be subjects related to censorship. Nevertheless, the suggested censorship type seems to be fair from European Morality and reputedly pushes society towards welfare and prosperity. These subjective characteristics were selected to identify the "Harmful and Illegal content blocking" censorship type and used to distinguish political and ideological censorship as a different type. Otherwise, there is no much sense to identify political and ideological censorship as a different type as it is mostly regulated by legislation that might label any content as harmful and illegal from a censor perspective.

2.1.2 Political and ideological censorship

This censorship type inevitably connects with the suppression of fundamental freedoms, such as freedom of expression, which is declared a fundamental human right by the Universal Declaration of Human Rights [23] and recognized by numerous international and states laws.

Arguably, the annual freedom on the Net report [24] by Freedom House delivers the most comprehensive observations of this type of censorship. In 2020, the report reveals that many states used the COVID-19 situation as an excuse for censorship engagement and blocking independent websites. Alongside independent websites blocking state officials and their affiliate supporters distributed disinformation to distract citizens from ineffective political decisions. Another trend of 2020 is abusive Surveillance for the benefit of public health. The report shows that in more than 30 countries, authorities are using the pandemic to establish mass surveillance in partnership with communications providers and other companies.

Although disinformation and mass surveillance become notable features in 2020, traditional political censorship cases also increased in their popularity. Critical and independent discussions, especially on political, social, and religious topics, provoke censorship measures against independent online platforms or particular persons. These measures might include

- Permanent or temporary shutdown of access to social media and other online resources
- Disconnection of the Internet or mobile networks
- Deployment of paid pro-government web commentators to manipulate online discussions and to influence public opinion
- Imprisoning, arresting, or assassinating individuals for their online activities

For more than a decade, China remains the leader of this censorship type. Since 1998 the Chinese authorities have developed and maintained the globally recognized censorship platform knowns as the Great Firewall (GFW). Moreover, in 2020 China has earnestly demonstrated leadership in mass surveillance thanks to the new machine learning and AI technologies.

2.1.3 Commercial censorship

The previous two sections deal with blocking content as required by law. However, there are two more common types of network resource blocking. The first type, which is used extensively, relates to preventing and responding to network security threats. For instance, most businesses put firewalls and Intrusion Detection Systems (IDP) as a barrier to cybercriminals' actions and malware infiltration. Furthermore, many Internet Service Providers (ISPs) block malicious traffic originating from their networks. This traffic often comes from compromised client's devices such as webcams and routers. Another example relates to email filtering, which is almost ubiquitous. It includes blocking unwanted bulk messages (spam) and malicious letters, e.g., phishing messages.

The second type of blocking is a network usage regulation. In this case, the blocking of content is driven by requirements for network utilization and workforce time management rather than combating certain types of data. Notably, employers frequently restrict their employees' access to social media in the workplace. Internet providers might block or allow access and regulate bandwidth to certain content depending on the services purchased by the customer. This type of censorship is rarely legislatively regulated, except for anti-competitive and net-neutrality rules. These rules impose equal treatment for all Internet traffic and prevent traffic discrimination by blocking or throttling [25].

2.2 Methods of online censorship

This section aims to observe online censorship methods regardless of their ethical, legal, or social aspects. Every method is assessed from a technical perspective, the type of data it can block, and an evaluation of possible negative implications of its application, so-called collateral damage. Thus, the suggested division is based on technological features only and not on the application practices by different blocking actors. Since a censor might apply every method separately or engage them simultaneously and combine with legislative measures to achieve better performance, the non-technical division would complicate the analysis.

2.2.1 Content removal and cyber sovereignty

Before observing technical methods of network access restrictions, let us discuss the most apparent blocking method - content removal. This method might be considered a promising approach, as it does not require substantial technical efforts and competence. However, the content removal method works well for centralized networks but fails in the global environment. Since the decentralized nature of the Internet challenges censors' endeavors to control online content, the concept of cyber sovereignty was adopted by several states.

Cyber sovereignty might be defined as a state's moving towards more closed and centralized control of the regional part of the Internet. In practice, this move might mean reestablishing government control over cross border gateways, as it was in Russia in 2014 [26]. Such countries as Russia and Iran aim to build their local versions of thoroughly supervised and independent national networks knowing as "Sovereign Internet" and "National Intranet" accordingly[24].

From a technical perspective, to create a national version of the Internet, in addition to full control of international traffic gateways, a local regulator should establish the country's own Domain Name System (DNS) root servers and procedures for distribution of domain names and IP-addresses. Local DNS infrastructure and IP distribution allow effectively cutting of a regional network and preventing global connectivity.

Because of the apparent collateral damage connected with the global network cut out, total cyber sovereignty is the last resort method. It might be used episodically to isolate some regional network segments temporarily. However, in the modern world, isolation equals stagnation and degradation; thus, this extremely effective censorship method has shallow practical capabilities.

2.2.2 IP address blocking

IP-address is a 32 bits (IPv4 address) or a 128 bits (IPv6 address) number that uniquely identifies a network device directly connected to the Internet. Groups of IP-addresses known as subnets are coordinated and allocated by Internet Assigned Numbers Authority (IANA). Both types of IP addresses are allocated in a tiered fashion. Users get their IP addresses from Internet Service Providers (ISPs). ISPs get their IP addresses or subnets from local, regional, or national Internet registries[27]. This hierarchical allocation scheme bounds IP addresses and geographical locations together. Thus, a censor might disrupt connectivity with a specific country or region by blocking subnets allocated to this territory.

Data transmitted over the Internet is divided into packets. Every packet contains addressing fields with IP addresses of its destination and source. These addressing fields might be used to block incoming or outgoing connections from or to particular IP-addresses or subnets. A firewall is the most common way to implement such filtering. In principle, firewalls are used by network administrators to monitor all incoming and outgoing packets and block those which contain blacklisted IP addresses or subnets.

Although a firewall can easily block a particular IP-address, this method can lead to overblocking as many services share the same IP-address, i.e., websites at the same hosting provider or a server behind Content Delivery Network (CDN). Furthermore, the modern Internet's dynamic nature in many cases allows fast IP address changing, i.e., in the case of a cloud environment.

2.2.3 DNS manipulation

Domain Name Systems (DNS) is a hierarchical system responsible for a human-readable domain name resolution into an IP address. When users connect to a particular website and enter the site's domain name in the browser address bar, they first initiate a DNS request that asks for the IP address corresponding to the site's domain name from users' system default DNS servers. The default DNS servers usually belong to the user's ISP. Thus, ISP has full control over domain name resolution and can provide incorrect DNS replies or no replies every time a user tries to resolve a blocked site's domain name.

Even if the user changes its default DNS servers, a censor might exploit an unencrypted nature of DNS traffic to modify independent DNS responses in order to return wrong or empty DNS replies. Since the censor typically does not differentiate end-users DNS traffic and traffic between DNS servers, DNS replies modification can lead to incorrect DNS entries for third-party DNS resolvers, thus cause collateral damage. Such behavior is known as DNS spoofing or DNS cache poisoning[27].

2.2.4 Keyword filtering

With unencrypted traffic, such as HTTP, a censor can block communications based on their content, for instance, by searching and isolating specific keywords in the traffic flow. Upon detecting the blocked keyword, the communication might be disrupted utilizing dynamic firewall rules or by TCP reset attacks[28]. Since more than 90% of today's web traffic is encrypted[29], this method lost its significance in its original implementation.

However, the censor might force users to install a censor-issued trusted CA certificate in their systems, thus gaining an ability to conduct traffic interception via Man In The Middle (MiTM) attack[30]. This method works as described below. A censor pretends to be a legitimate website and issues a bogus certificate with the censor's public key for this site. Under normal circumstances, the censor cannot sign its forged certificate with any trusted third-party Certificate Authority (CA) and cannot use the forged certificate as browsers will notify a user about the MiTM attack. However, the censor can force users to install the censor's root CA certificate into their systems. Thus, the system will trust the censor's bogus certificate. Using this technique, the censor can imitate any website, intercept users' traffic, alter the website's content, and apply keyword filtering to encrypted connections.

2.2.5 TLS filtering

Transport Layer Security (TLS) protocol encrypts the content of a communication, except the domain name of the requested resource, which is defined by the Server Name Indication (SNI) filed in unencrypted client Hello message during TLS handshake process[31]. As a censor can still observe the destination domain, it can disrupt the TLS handshake or negotiation process, hence blocking the communication. The newest TLS protocol version 1.3, which was introduced in 2018, added support for encrypted SNI and seriously limited[32] the possibility of TLS filtering.

2.2.6 Traffic types and protocol blocking

TCP or UDP port filtering is the most affordable method to block a particular protocol. For instance, the traditional HTTP uses TCP port 80 and can be blocked by preventing connections to this port using a firewall. Although web traffic can be blocked in this way, the blocking might be avoided simply by port changing. For this reason, a censor can utilize Deep Packet Inspection (DPI) as a more advanced alternative to port filtering.

DPI blocking involves the installation of content filtering devices between the end-user and the Internet, known as a Network Intrusion Prevention System (NIPS). Such systems responsible for traffic patterns evaluation and isolation of keywords, application types, or separate files. Effective DPI blocking requires pre-defined signatures or trained machine learning algorithms. Such traffic features as protocol keywords, packet sizes, bandwidth deviations, and file names can be used to produce signatures and train machine learning algorithms.

In a few words, DPI operates the following way. As a first step, all the packets coming from and to the supervised network interface (or wiretap) are collected and generally represented as traffic flows using hash tables, then stored for further processing. With the next step, the DPI engine examines the stored packets payloads and traffic flows for recognizable patterns using regular expressions, finite automata, or machine learning algorithms[33]. As a final step, traffic processing decisions can be made automatically or manually based on performed DPI analysis.

DPI is effectively used to block specific applications and data exchange, such as peer-to-peer file-sharing and Voice over Internet Protocol (VoIP). However, this type of blocking is computationally intensive; thus, it might be costly. Furthermore, the vast number of network applications and the ubiquitous use of traffic encryption challenge the DPI analysis and reduce its effectiveness, resulting in a high number of false-positive and false-negative cases, which eventually lead to collateral damage.

2.2.7 BGP routing and network disruption

Internet backbone relies on a Border Gateway Protocol (BGP), a routing protocol responsible for routing between Autonomous Systems (AS). A censor might disrupt this global routing mechanism for its users by advertising wrong subnets or IP-addresses. Thus, effectively block its users from reaching all or some destinations outside their AS. This technique is known as BGP tampering, also sometimes called null-routing. This method can block any number of destinations from single IP-addresses to almost all available address spaces. Thus, it is highly effective as a temporary network shutdown mechanism in a particular country or region. However, it comes with severe collateral damage due to the obvious economic and social consequences.

2.2.8 Bandwidth throttling

A censor might intentionally reduce the quality of service for different traffic types, i.e., by slowing down the connection to particular websites or applications. Since this blocking method is challenging to prove and identify, users might search for an issue with their Internet connections. Fortunately, the method's effectiveness is limited because the connection is not entirely restricted.

2.3 Circumvention solutions and strategies

This section collects the description of the most common circumvention strategies used to thwart the censorship methods described above. The section also observes the mitigating potential and a circumvention solution effectiveness against suggested censorship methods, together with a general description and principles behind a particular approach. As usual, suggested strategies can be combined to achieve better circumvention potential. However, even separately, each mitigating solution provides a substantial privacy and surveillance protection level if used correctly. All solutions are divided into five categories based on generally accepted and currently used classification by network professionals and researchers.

2.3.1 Secure DNS

Because of the unencrypted nature of conventional DNS protocol, changing user's default DNS servers is an essential task to maintain DNS queries' privacy and security. Anyone can use external DNS providers such as Cloudflare (1.1.1.1, 1.0.0.1) or Google (8.8.8.8, 4.4.4.4) unless their IP-addresses are not blacklisted, but even in this case, plenty of other options exist. Furthermore, it is vital to use an encrypted version of the DNS protocol to avoid possible DNS spoofing attacks.

Currently, two versions of encrypted DNS are broadly recognized. These are DNS over TLS (DoT)[34] and DNS Queries over HTTPS (DoH)[35]. Both protocols are based on Transport Layers Security (TLS) and serve the same purpose of DNS traffic encryption.

DoT encapsulates the original unencrypted DNS protocol into TLS traffic flow and uses 853 TCP port, the same way as it was done for other legacy protocols such as HTTP, IMAP, and SMTP. Usage of a separate port is the shortcoming of such a straightforward approach as the port can be blocked by a firewall. Consequently, the user might have to use the conventional version of DNS (TCP/UDP port 53), allowing the censor to supervise DNS requests[36]. Such attacks have their own name, SSL Downgrade or SSL Stripping attack.

The second version of encrypted DNS (DoH) is resistant to possible SSL Downgrade attack as it uses HTTPS (TCP port 443) as a transport for DNS queries. HTTPS makes DNS queries almost invisible as there is no known way to discover what type of content is hiding inside HTTPS encryption. Furthermore, HTTPS allows applications to use existing browser APIs to initiate DNS queries[36]. To summarize, both versions of encrypted DNS combined with independent DNS resolvers thwart DNS possible manipulations and protect users from possible DNS spoofing attacks.

2.3.2 Proxies

A proxy generally might be considered an intermediate server between a client and a destination, relaying incoming data from the client to the destination and vice versa. Therefore, the traffic relaying might be used to bypass the blocking, as the client does not connect directly to the destination.

The proxies classification includes HTTP(S), SOCKS, and encrypted proxies, such as Shadowsocks. Proxy support is an internal feature of the HTTP protocol. RFC 7230[37], related to HTTP1.1, describes proxy as a "message-forwarding agent" that receive requests from a client and "satisfies those requests via translation through the HTTP interface." RFC7230 also defines a CONNECT method that is used by a client to establish a proxy tunnel. Otherwise, the HTTP proxy connection does not differ much from the standard HTTP connection; thus, it is susceptible to the same circumvention methods as HTTP. In addition, the HTTP proxy connection can be secured over TLS the same way as it is used for HTTPS. The second version of HTTP also supports the HTTP connection tunneling and uses the same CONNECT method to establish a proxy tunnel over a single HTTP2.0 stream[38].

RFC1928[39] describes SOCKS protocol as a more general proxy protocol and, in contrast with HTTP proxies, SOCK supports tunneling for arbitrary TCP or UDP traffic. Although SOCKS supports authentication, it does not provide inherent encryption for tunneling traffic. Therefore, the original implementation might be easily blocked through DPI as the SOCKS protocol can be easily recognized in the traffic flow. To overcome this flaw, some developers added encryption support for their SOCKS implementation.

A notable example of encrypted SOCKS proxies is Shadowsocks[8]. It is a secure SOCKS proxy software with Authenticated Encryption with Associated Data (AEAD) ciphers support. Such ciphers enable confidentiality, integrity, and authenticity for SOCKS protocol and thwart censorship by encrypting the entire proxy tunnel.

Additionally, proxies might be divided into forward and reverse proxies. Forward proxies provide tunneling service to a client and might be used to bypass censorship and anonymize its identity by hiding the client's IP-address. The reverse proxies might be used to hide the identity of a server from a censor. They also serve load balancing, caching, and compression purposes and usually used by CDN networks. The circumvention software might hide censored connections with reverse proxies using the so-called domain fronting technique[6].

2.3.3 Virtual Private Networks

Virtual Private Network (VPN) is another network tunneling mechanism that can be used to bypass censorship. Unlike proxies, VPNs always provide integrity, authentication, and confidentiality for the encapsulated traffic. Nowadays, dozens of VPN protocols exist, and most of them can be used (to some extent) for censorship circumvention purposes. Arguably, IPSec, OpenVPN, Wireguard, and SoftEther represent the most notable VPN protocols. However, this list is neither exhaustive nor concise enough to discuss all the items in this study. Thus, this section is limited to a brief description of OpenVPN and Wireguard, the two most promising VPN technologies in 2020.

OpenVPN protocol enables tunneling of IP packets or Ethernet frames over a specific UDP or TCP port; in other words, OpenVPN encapsulates IP or Ethernet into TCP or UDP protocol[40]. It uses a TLS-like approach for encryption, authentication, and integrity. The first version of OpenVPN was introduced in 2001 and quickly became popular as it was one of the few cross-platform and opensource VPN solutions at those times. Initially, it was a simple point-to-point IP over UDP tunnel with Blowfish cipher support and SHA1 HMAC authentication[41]. Later, OpenVPN developers added the support for X.509 certificates, smart cards, and a two-factor authentication mechanism. OpenVPN utilizes TUN/TAP virtual network interfaces to route all traffic through a tunnel and does not require application-specific configuration such as entering IP-address and port for proxy servers. Since OpenVPN supports full encryption of TLS handshake using a pre-shared key (TLS crypt) to hide the TLS negotiation from eavesdropping[42], it might be considered a censorship-resistant protocol.

In contrast with OpenVPN, WireGuard is a relatively new protocol. It was developed in 2015 and has already got its popularity thanks to the unprecedented performance and implementation simplicity. Since the early beginning, WireGuard uses state-of-the-art ChaCha20 and Poly1305 AEAD cryptography and encapsulates IP packets into UDP[43]. Unlike OpenVPN and IKE, WireGuard does not care about key distribution and uses an authentication model similar to SSH, i.e., client and server authenticate each other with their own public keys[43]. WireGuard uses its handshake implementation and does not rely on standard TLS. It was designed not to respond to unauthenticated packets, thus gaining its resistance against service discovery attacks. Unfortunately, the protocol's handshake has numerous unique features, i.e., the handshake happens every several minutes to provide keys rotation for perfect forward secrecy. Thereby, the protocol handshake's unique characteristics facilitate DPI analysis and make it possible for an adversary to discover and suppress WireGuard's connections.

These two examples illustrate the importance of concealing protocol handshake detail from DPI analysis. Despite being an excellent protocol, Wireguard fails to hide its handshake negotiation from eavesdropping, hence fails to resist against DPI blocking[44]. The DPI resistance, however, was not the intention of the developer. A similar situation was with OpenVPN until the tls-crypt option, which provides the additional layer of protocol encryption, was added in 2018 with OpenVPN 2.4 [45]. TLS-crypt made it possible to hide OpenVPN from DPI engines. This study's scope does not provide a chance to present a detailed analysis of how particular VPN protocols stand against blocking. Nevertheless, the example of OpenVPN illustrates the primary approach employed by developers to surpass blocking attempts. It must be noted that feckless blocking of VPN protocols leads to severe collateral damage as most VPN solutions are used to interconnect business networks or provide remote access for employees.

2.3.4 Network tunnels

This category includes such circumvention strategies as SSH and RDP tunnels. These tunnels are based on SSH and RDP protocols, which are standards for secure access to remote servers. SSH tunnels enable transportation of arbitrary traffic over a secured SSH connection using port-forwarding to redirect any TCP port to a tunnel[46]. Thus, SSH tunneling provides an additional encryption layer, a VPN-like ability to access internal services from outside networks, and a way to circumvent censorship using a widely used and most popular remote access protocol. RDP tunnels follow the same principle and might also be used to bypass censorship. Since the blocking of SSH or RDP results in totally unacceptable consequences for most censors, SSH and RDP throttling is the most obvious way to reduce the circumvention performance of tunnels.

2.3.5 Specialized circumvention software and protocols

Increasing censorship capabilities challenged the traditional VPN and proxy protocols and accelerated the development of dedicated anti-censorship solutions. Such protocols as obfs4, Lampshade, MTProto, and obfuscated SSH were developed and used in Tor, Lantern, Telegram, and Psiphon to provide desirable blocking resistance.

Obfs4[39] establishes an additional obfuscation layer for Tor bridges via Pluggable Transport (PT) interface[47]. To use obfs4 transport, a user must know the obfs4 node unique ID and Curve25519 public key distributed in an out-of-band fashion. To connect to the obfs4 node, a client initially sends its Curve25519 public key, some random data, HMAC calculated over the server's key, previously obtained unique node ID, and one more HMAC with a timestamp[39]. The client's public key is encoded with Elligator[48] obfuscation to become indistinguishable from arbitrary data. The obfs4 node only responds if the received HMACs are calculated based on the node's public key and ID. The obfs4 node does not respond if the client sends incorrect HMACs and closes the TCP connection after a random period of delay[11].

MTProto[7] obfuscates Telegram connection to mitigate blocking in certain countries. Similar to obfs4, the secret key is distributed in an out-of-band manner. The key is derived from the hash of a random seed and then encrypts a particular magic value. The server, upon decryption, expects to obtain the client's magic value. If it is not the case, the server takes no action and keeps the TCP connection[11] forever.

Lampshade and obfuscated SSH exploit the similar approach of out-of-band secret key distribution and not replying to invalid protocol messages[11]. This approach delivers necessary resistance to DPI analysis and possible blocking as all communications are encrypted and usually obfuscated starting from the first packet. Although some protocol features, like connection reset delay, might still be used to discover and block described solutions, these flaws are usually fixed with new versions.

2.4 Summary

This chapter describes online censorship and divides it into three major types: harmful and Illegal content blocking, political and ideological censorship, and commercial censorship. It provides valuable examples and distinctive characteristics of these censorship types. The chapter further explains censorship methods from a technical perspective and possible collateral damage connected with each blocking technique. Eventually, the chapter summarizes the most notable mitigating solutions such as proxies and VPNs and illustrates their mitigating potential.

3 The Great Firewall

This chapter provides an in-depth observation of the Great Firewall of China and underlines technologies behind its operations, such as a TCP reset engine and active probing. Additionally, an overview of the Chinese segment of the Internet is provided to show a general picture of nationwide blocking.

In 1996, the Chinese Ministry of Public Security initiated the "Golden Shield Project" to establish a higher level of control of information sources. Control over the Internet was one of the main goals of the project. This aim was addressed to the emerging Great Firewall of China (GFW)[1] – a surveillance and censorship tool that filters cross-boarding connections and blocked data restricted by the Ministry of Public Security. Almost 15 years later, the GFW plays a crucial role in Chinese Internet censorship by blocking access to numerous foreign websites, such as Facebook, Google, and Wikipedia. Today, the firewall is a sophisticated yet ambiguous piece of software because its internal mechanisms are hidden from research communities.

Many academic and civil activists have spent significant efforts in studying and analyzing the firewall. Previous studies showed that the GFW employs a whole set of censorship strategies, including passive on-path traffic inspection[49], TCP reset attack[28, 49], BGP tampering[50, 51], DNS injection [52, 53, 54], TLS Server Name Identifier (SNI) and Encrypted SNI (ESNI) filtering [5], active probing[4, 3, 2, 55], Quality of Service (QoS) degradation [56], and machine learning [10, 57] in order to recognize unwanted traffic patterns. This arsenal relates only to the technical part of the GFW's censorship, however legislative questions and self-censorship are out-of-the-scopes in this study. The firewall functionality was divided into several subsystems described further and illustrated by Figure 1.

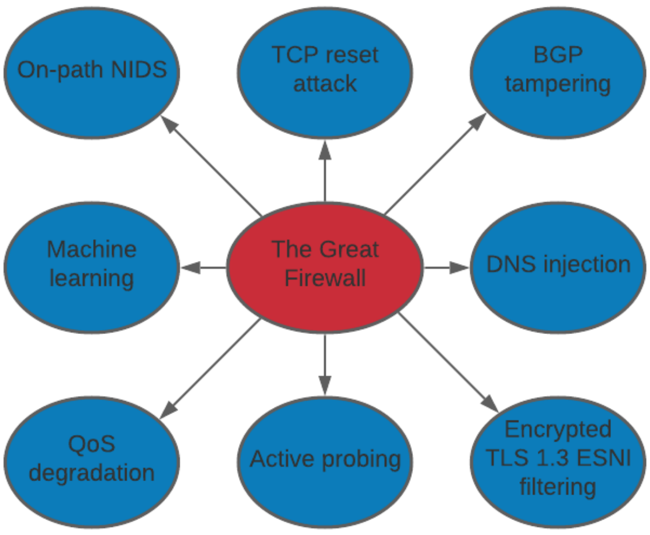
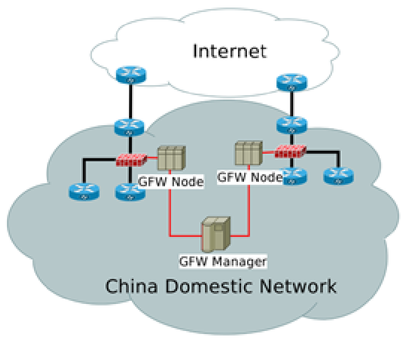


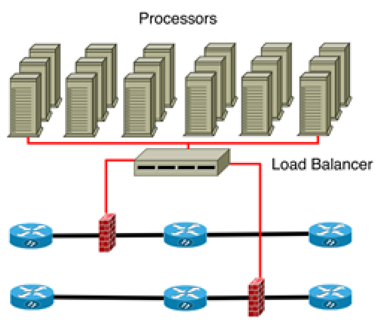
Figure 1: The firewall capabilities.

3.1 The overall architecture

In [52], the overall architecture and structure of the firewall were studied. The study revealed the firewall's node locations and their internal structure, the usage of load balancing, and centralized management. The firewall was observed as a centrally managed on-path Network Intrusion Detection System (NIDS) with DNS Injection, and TCP reset capabilities. The GFW manager rules GFW nodes, which operate in clusters and analyze connections coming through traffic exchange points. Thus, nodes work not only at the network edge but within the inland part of the Chinese Internet. Namely, nodes are responsible for traffic sanitization, and they are the main executors of all previously mentioned filtering methods.



(a) Overall Architecture



(b) Structure of one GFW Node

Figure 2: On-path nature of the GFW, overall architecture and nodes load balancing[52].

3.2 Firewall's mechanics

The GFW utilizes almost all known censorship methods and some unique mechanics such as active probing and ESNI filtering. The described mechanics, together with hierarchical blocking infrastructure, implement most censorship methods from Chapter 2 and can reliably block domain names, IP-addresses, and many circumvention tools.

3.2.1 Passive traffic inspection and TCP reset attack

It has been reported that the firewall operates as a passive on-path network intrusion detection system (NIDS)[49]. Similar to other NIDS systems, the GFW passively monitors traffic and disrupts unwanted connections by TCP reset attack[28, 49, 53]. TCP reset attack exploits the reset mechanism of TCP protocol. According to RFC793, If a communicating party needs to reset a TCP connection, i.e., in case of a wrong ACK field, it can send a packet with a reset (RST) flag. This feature of TCP implementation is used by the GFW, which sends forged TCP reset packets (RST type1 or RST/ACK type 2) to communicating parties with the intention of connection termination. The firewall sends forged reset packets continuously for about 90 seconds[58]. Any SYN packet between endpoints triggers a forged RST/ACK during this time interval, and any other packets trigger a forged RST reply[58]. Figure 3 demonstrates the ongoing TCP reset attacks.

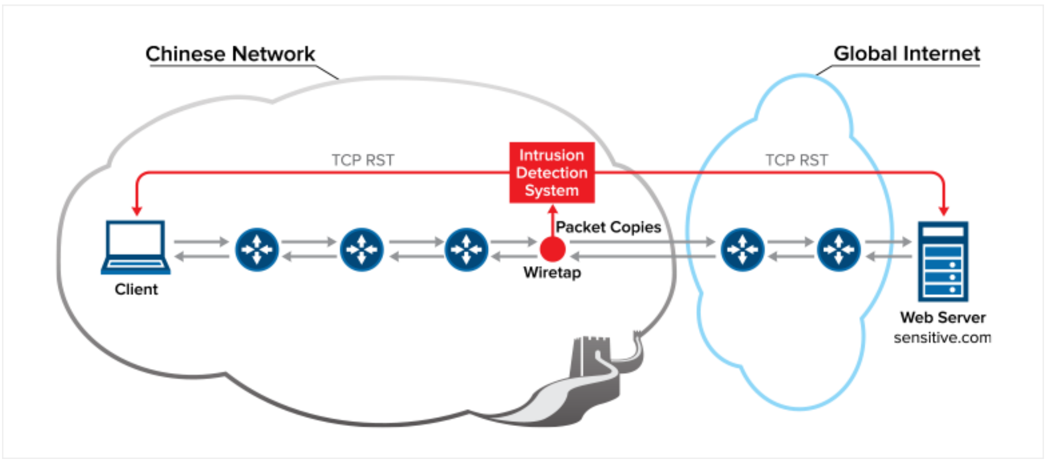


Figure 3: The GFW's TCP reset attack [58].

Since the GFW is an on-path system , it passively monitors all cross-border connections. It can inject additional packets into any traffic flow, but it cannot prevent packets that are already in transit from reaching their destination. Unlike traditional in-path firewall systems, this approach does not lead to a connection slow down and allows monitoring much more traffic with less computation power consumption. Figure 2 additionally demonstrates the on-path nature of the GFW.

In 2009, Weaver [28] developed an efficient detector for forged TCP reset packets exploiting the race conditions that out-of-band RST injectors fundamentally face. Furthermore, due to its on-path nature, the firewall cannot directly manipulate traffic, thus cannot prevent packets that are already in transit from reaching their destination. Both Inability to block connections immediately and Weaver's detector made it theoretically possible to bypass the firewall censorship by rejecting forget reset packets. Unfortunately, the censor had resolved this vulnerability by employing BGP tampering.

3.2.2 BGP tampering

BGP tampering, also known as BGP hijacking or null routing, is used by the firewall for IP address blocking. It is a straightforward, lightweight, yet extremely efficient filtering method that utilizes BGP routing properties. Thus, the method might only be used by organizations with an Autonomous System (AS) status.

According to RFC1930 [59], an Autonomous System is a connected group of one or more IP prefixes run by one or more network operators with a single and clearly defined routing policy. . When an adversary AS advertises a route to a subnet that it does not manage, that advertisement, if not filtered, can be propagated and be inserted into the routing tables of BGP routers over the Internet. Until someone detects and corrects the routes, traffic to this subnet will be forwarded to the adversary AS. To perform a successful hijack, the adversary must advertise either a more specific route using a smaller IP prefix than a legitimate AS announcement or announce a shorter route than legitimate AS. Furthermore, full control over the BGP operations must be achieved to announce BGP routes and utilizes BGP hijacking. It is not an issue, however, for nationwide network operators and governments.

The GFW peers with border routers of all Chinese ISPs and injects blacklisted IP addresses into their BGP routing tables, thus null-routing or hijacking all traffic coming to blacklisted destinations [51, 50]. Although null-routing affects only outgoing traffic, it disrupts cross-border communication by breaking the three-way handshake procedure.

BGP tampering effectively blocks blacklisted IP-addresses. However, the effectiveness highly depends on the blacklist's accuracy and completeness. The blacklist should be continuously maintained and updated. Furthermore, the spread of cloud technologies reduces the efficiency of this method, as the IP-address can now be changed in a few seconds. Another limitation is the possible collateral damage; the IP-address blocking might disrupt innocent resources if they share the same IP address with the filtering target. Thus, together with BGP tampering, the firewall uses more sophisticated censorship techniques.

3.2.3 DNS injection

In 2002, it was spotted that the GFW started to inject DNS responses [60]. The GFW uses a well-known technique of DNS injection to restrict the number of available web sites. Many studies [52, 54, 53], have shown that the filter injects forged DNS in replies to DNS queries containing prohibited domain names and keywords. Injection happens regardless of a DNS request type; both iterative and recursive DNS requests are replied with forged records. Additionally, the firewall does not differ request to/from recursive resolvers (cache servers), Authoritative name servers, TLD Servers, and Root DNS servers. A request is blocked if it contains a prohibited domain name or keyword.

Fortunately, such projects as hikinggfw.org and greatfire.org constantly monitor firewall's censorship and compile a list of blocked domain names and websites. According to greatfire.org, in October 2020, 155 of 1000 Alexa top sites are blocked in China, including google.com, youtybe.com, facebook.com, Wikipedia.org, reddit.com, and many others. Moreover, the total numbers of 14495 keywords and 35332 domains have been reported to be censored by the GFW in 2014 [52]. At time of this writing (October 2020), the firewall blocks 981352 domains and 3086 keywords [61]. A recent study [54] about censorship among Alexa top million domains has revealed a 2.8% increase of censored domains over nine months from September 2019 to May 2020. Moreover, it has been reported that every 1 of 23 DNS request is forged [54].

Unfortunately, Chinese DNS injection applies not only to Chinese users but sometimes influences global communications. In [52], it was shown that the injection method also works for third-party DNS resolvers from outside China, as the firewall always reacts to all queries originated from the country. Moreover, [53] showed that injections are triggered both by inbound and outbound traffic, which results in collateral damage due to the poisoning of the DNS cache of innocent DNS resolvers. Sparks [53] demonstrated that such factors as iterative queries type, name servers hierarchy, server distributions, dynamic, and anycast routing might cause DNS requests to transit via censored network even when both communication parties are outside the network border.

The Domain Name System Security Extensions (DNSSEC) seems to be a promising mitigating solution. However, the slow adoption rate of new internet technologies, i.e., no deployment of IPv6 in China, still provides much space for DNS tampering.

3.2.4 TLS SNI and ESNI filtering

Historically, the GFW performed HTTP requests filtering based on a pre-defined keywords list [1, 62]. Nevertheless, the spread of SSL/TLS HTTP (HTTPS) stopped this practice as web traffic is generally encrypted nowadays. Since TLS provides authentication and encryption, web traffic cannot be read or tampered by a censor. Although TLS protects the content of transmissions, it does not hide a destination party. The destination host is defined by Server Name Indication (SNI) field in a TLS handshake. Chinese censor has used this filed to perform HTTPS censorship and prevent users from reaching blocked web sites [32].

In response to SNI filtering, the Encrypted SNI was proposed for TLS 1.3 in order to thwart the TLS censorship. However, ESNI introduction triggered a new arms race round among censorship and circumvention technologies. Chinese censor's reply to the introduction of TLS 1.3 encrypted ESNI was straightforward. The GFW started to block ESNI capable connections in July 2020 [63]. Later this finding was confirmed in [5], together with some technical insights such as blocking mechanism details and blocking duration.

3.2.5 Active probing

Active probing is a simple but yet powerful method of circumvention server identification that is widely used by the firewall[4, 3, 55, 2]. The GFW engages active probing when it cannot reliably identify the connection type and prove that it is used to bypass censorship and associated with a hidden proxy or VPN protocol. In such an ambiguous situation, the firewall sends a probe, an artificial handshake request, to a suspicious server. The GFW may confirm the circumvention nature of the server by observing the server reply to the probe request. If the server response corresponds to a known circumvention tool, all further communication attempts are blocked.

Typically, active probing operates like a reactive network scanner triggered by ambiguous traffic inspection. The GFW may perform proactive and idle probing; however, there is no clear evidence of idle probing. It has been shown that probing is successfully used against Tor, SoftEther, Obfs2, and Obfs3 protocols and proxies that provide access to the Google app engine using domain fronting[6].More detailed observations of probing activity against Tor network are provided in the corresponding chapter.

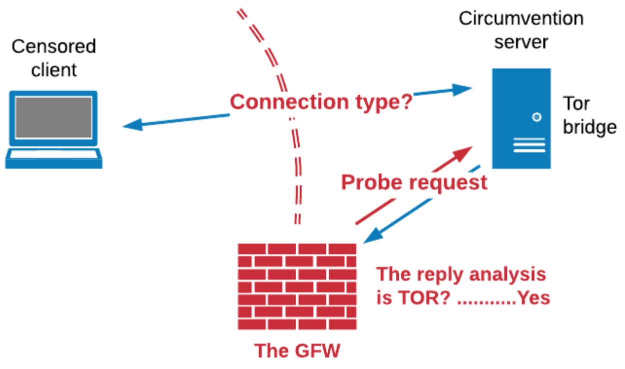


Figure 4: Active probing process.

3.2.6 QoS degradation

There are not many scientific articles about circumvention connections QoS degradation by the GFW. Nevertheless, some clear evidence of such firewall behavior is presented. Bevand [64] revealed that the firewall slows down SSH SOCKS tunnels and induces the packet loss around 70-80% after several minutes of the tunnel being actively used. He also noted the same problem for Polipo web proxy and proxy over TLS connections. In addition, he proves that the GFW uses side-channel leaks [65] in TLS, for example, by observing packet sizes.

3.2.7 Machine learning

Although there is no direct evidence of machine learning usage by the GFW, it makes sense that the GFW staff utilize machine learning and train traffic classification models based on actual circumvention tools setups. Thus, the firewall knows the characteristics of different circumvention protocols and can identify them automatically. The vast number of traffic classification studies by Chinese authors indirectly supports this claim [66, 67, 68, 57, 69]. Since the internal mechanisms of the firewall are hidden, it is challenging to precisely estimate to what extent the GFW uses machine learning and traffic classification algorithms.

3.3 The Chinese Internet

Even if the firewall appears to be a single entity, its components and subsystems are distributed across major Chinese ISPs and operate independently from each other, sharing common blacklist data and software[49].It happens because of the Chinese ISP market's remarkable characteristic, where the state owns all routers, and most ISPs rent bandwidth from government-owned China Telecom. Historically, the Chinese Internet included only four major nationwide ISPs, specifically CSTNET, China telecom, CERNET, and CHINAGBN. Nowadays, six additional nationwide providers are operated[70], including:

– China Academy of Information and Communications Technology

– China Telecom

– China Mobile

– China Unicom

– China Radio and Television Network Co

– CITIC Networks Co.

A small number of ISPs show that the Chinese Internet is highly centralized with no space for fair competition. The biggest providers, such as China Telecom and China Unicom, act as a monopoly. They dictate conditions to smaller operators, establish protecting interconnection prices, thus preventing small operators from being profitable[71].

Current interconnection agreements lead the dominant operators to have more market power than competitors. Small operators cannot compete with China Telecom and China Unicom and often run out of business [72]. Hower, July 1, 2020, the Chinese government turned toward a fair interconnection market settlement and introduced a new regulation that aimed to discontinue protecting interconnecting practices[70]. The new regulation supports mobile operators, especially China Mobile, the most prominent Chinese operator with 180 million users.

China and the rest of the world are connected via ten traffic exchange points and communicate through a limited number of submarine cables. In 2015, there were only three traffic exchange points (Beijing, Shanghai, Guangzhou) connected to the global Internet[70]. Since 2015, seven new exchange points have been added in Chengdu, Wuhan, Xi'an, Shenyang, Nanjing, Chongqing, and Zhenzhou. These were national level exchange points before, and they were promoted to international traffic gateways[70]. Nevertheless, ten internet exchange points appear to be insufficient for a country with more than a billion citizens.

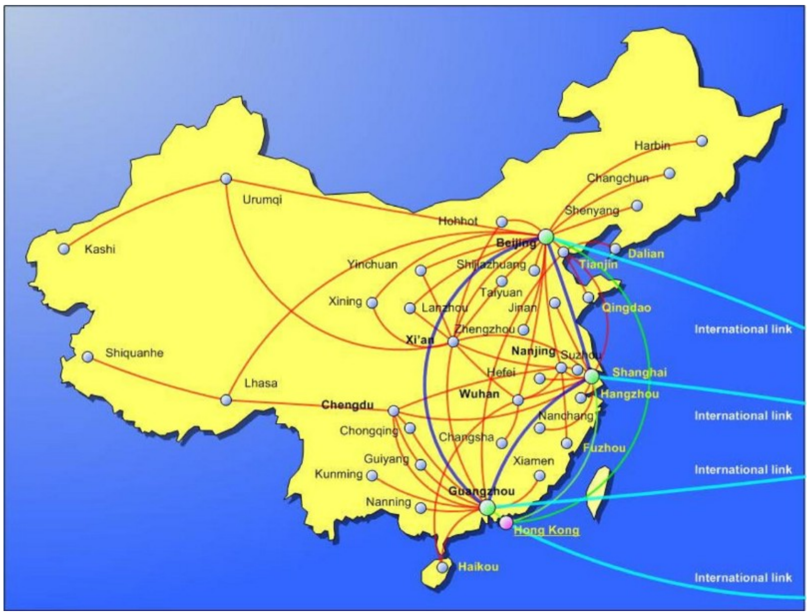


Figure 5: Chinese global Internet gateways in 2015 [70].

Since the state owns all Tier 1 infrastructure, the most sophisticated and the final filtering happens on the Tier 1 level. Moreover, the first filtering level (Tier 2 operators) mostly responsible for DNS censorship and HTTP(S) traffic purification[58].

The network complexity grows with the increase of international traffic exchange points and the encouragement of competition. This growth challenges the Chinese censor and makes demands for blocking technologies described above.

3.4 Summary

This chapter observes seven known blocking mechanics of the GFW:

- Passive traffic inspection and TCP reset attack
- BGP tampering
- DNS injection
- TLS SNI and ESNI filtering
- Active probing
- QoS degradation
- Machine learning

It describes how the firewall utilizes every particular mechanic to block different content types and demonstrates the principles behind their realizations. Additionally, the overview of the firewall's overall tiered architecture is provided, revealing the firewall's on-patch nature and load balancing scheme. Finally, the Chinese Internet is observed to provide a broader view of the national telecom market and show how its structure facilitates censor's task.

4 The Tor anonymity network

Since the introduction of the GFW, Chinese users have sought a reliable way to bypass internet filters. In the 2010s, the Tor anonymity network was one of the most popular ways to circumvent the GFW. Unfortunately, Tor was rapidly blocked soon after the censor realized its circumvention potential. Thus, a continuous arm-race between Tor and the GFW began. Tor is a second-generation onion router anonymity network introduced by Dingledine et al. in 2003 [73]. Unlike the initial onion routing concept, Tor was practically oriented. From the very beginning, it supported features crucial for production implementation, including perfect forward secrecy, congestion control, directory servers, integrity checking, configurable exit policies, and practical design for hidden services via rendezvous points.

4.1 The overall architecture

The Tor network consists of onion routers (OR) or Tor nodes run by volunteers. In addition, every user runs an onion proxy (OP), which is necessary for end-user connections. Every Tor node is connected to all other Tor nodes, thus creating an overlay network over the Internet. By contrast, a Tor proxy is responsible for obtaining a network consensus from a Tor directory, building circuits across the Tor mesh, and acting as a SOCKS proxy for client applications. Such an application is most commonly a Tor browser, a specially constructed version of an anonymous browser based on Firefox, but it can be any other application with SOCKS proxy capabilities.

Building a circuit between Tor nodes is crucial to provide anonymity to users. Every circuit is an established connection path across onion routers. At least three routers are necessary to construct a circuit. The exterior router is called the exit relay or exit node, the intermediate router is termed the relay node, and the closest to a client router is known as the guard node. With each node in a circuit, the OP negotiates a symmetric key using a combination of RSA and AES encryptions[74].

As soon as the circuit across the Tor network has been established and the keys have been negotiated, the OP divides traffic into fixed-size cells and then encrypts the payload and the header using the exit node key. Next, it adds a new header with the exit node as a destination and then re-encrypts the payload and the header using the relay node key. After this, it adds a new header with a relay node as a destination, once again encrypts the payload and header with the guard node key, and adds the final header with the guar node address as a destination. Thus, an initial payload and a header are encapsulated three times and ready to travel across the circuit. While traveling through a node, the corresponding cell header is decrypted, so every node knows where to send it further. However, the full path is known only to the OP, whereas the OR knows only its neighbors within a circuit, thus achieving the necessary level of privacy. Figure 6 illustrates the process in detail.

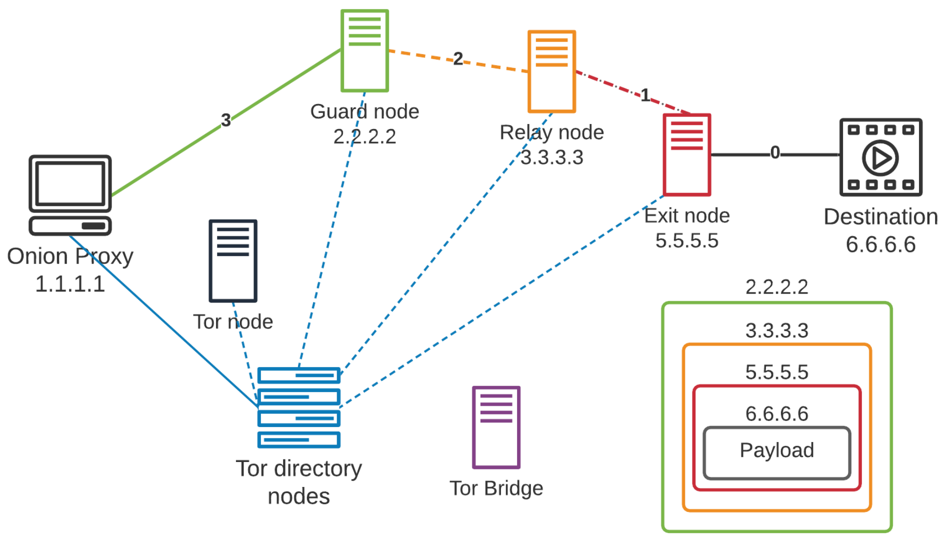


Figure 6: The Tor network components, a circuit establishment, and an encapsulated payload encryption.

4.2 How the GFW blocks conventional Tor

The original onion routing concept used a flooded update mechanism, similar to link-state routing, to inform mesh participants about all network nodes. Unfortunately, for the sake of simplicity, Tor developers decided to discontinue this practice. Instead, they introduced a centralized directory authority which consists of a small number of trusted routers. These routers trace changes in node states and mesh topology, i.e., the number of routers, their status, and adjacencies. Each of the ten directory servers operates as a web server and provides network state data and a network node list to the OP (clients) upon their request[73].

Moreover, the IP addresses of the directory servers are constant and hardcoded into the software (OP). Since directory servers are open and available for anybody, an adversary can disrupt regular Tor operation by filtering access to directory authority servers. Moreover, nothing prevents it from obtaining full network topology information, including the IP addresses of all Tor nodes. The next chapter provides additional evidence that all Tor directory servers are unavailable from within China as well as most vanilla Tor nodes.

4.3 Tor bridges for censorship resistance

It was reported that the GFW started to block the Tor Project website in 2008 [75], and by 2010 the filtering was established for a complete network [76]. Consequently, the Tor Project introduced Tor bridges, a new type of OR whose IP addresses are private, as their data are not available through the public directory authority. Bridges became the private nodes of the Tor network, and their primary purpose is to bypass censorship. Furthermore, they provide the same privacy protection against surveillance and traffic monitoring as regular Tor nodes.

4.4 Active probing and Tor

In 2011, right after their first usage from inside China, the first reports emerged that the GFW had begun blocking unpublished Tor bridges [77]. Eventually, the situation attracted the attention of the Tor Project developers and the research community. Later that year, Wilde [55] was the first who reported the dynamic nature of the new blocking method and performed an analysis of the blocking mechanism. He showed that the SSL ClientHello message, which is sent from a client in China to any Tor bridge outside China, triggers a new Tor handshake connection, a probe connection, to the Tor bridge from an unknown Chinese host. Wilde noted that probe connections arrived in 15-minute intervals after the first SSL ClientHello message. In addition, he proposed a way to confuse the probing mechanism by modifying the SSL ClienHello message. It was sufficient to change a list of available ciphers during SSL negotiation in order to distract probing. Unfortunately, this strategy worked for only a limited time and required constant changes to the Tor protocol specification.

Later observations by Winter and Lindskog [2] revealed that bridge blocking occurs based on an IP:port tuple and not an entire IP address. They showed that the GFW drops SYN/ACK packets arriving from the bridge to a client but not TCP SYN packets from the client to the bridge. They demonstrated that blocking is continuous as long as the firewall can probe the bridge; if it is unable to do so, blocking is removed in 12 hours. Their study proved that Chinese domestic traffic is not affected by Tor fingerprinting. They revealed scanner IP address distribution and showed that half of all probes came from a single IP – 202.108.181.70, whereas another half of the probes' IP addresses were almost uniformly distributed. By analyzing the difference in TTL, they proved that the GFW practices IP address spoofing, i.e., temporarily borrows legitimate users' IP addresses and uses them as probing sources. In addition, their work confirmed Wilde's findings of probing the timing and scanning queues of connections. Finally, they suggested a circumvention strategy based on packet fragmentation in order to complicate the task of traffic analysis.

In 2015, Ensafi et .al [3] performed observations of GFW probing activities by mining production web server (HTTP, SSH) application logs for a period of 2.5 years. In addition, they observed Chinese firewall interaction with a specially constructed infrastructure of Tor bridges and clients in a controlled experiment. Their analysis included an assessment of probing effectiveness against Tor bridges, analysis of probing triggers and delays, probes sources analysis, identifying different types of probes, and proposed fingerprinting technique in order to understand the physical structure of the firewall probing subsystem. Active probing effectiveness was reported to be about 98% against vanilla Tor bridges for connections from China Unicom and around 88% for CERNET clients. The TLS ClientHello message was confirmed to trigger the probing event; however, the 15-minute probing queue suggested by Wilde[55] was discontinued. Probes were found to operate in real-time, i.e., arrive within one second of the triggering event. Filtering was confirmed to be based on an IP:port pair. The Probes' sources were proved to be almost always unique, with only 5% of IP addresses used two or more times. The only outlier was the above-mentioned 202.108.181.70, with 248 out of 16083 observable probes connections. Although this value is significantly less than the observations made by Winter and Lindskog [2]. Probes were categorized into five distinct types. Three of these types were employed against Tor, obfs2, and obfs3 protocols, while others targeted SoftEther and Google App Engine. It has been suggested that despite probes' IP address diversity, only a minor number of "independent processes" are operating them. This conclusion originated from an analysis of TTL distribution, MSS patterns, ISN numbers of SYN segments, source ports, and TCP timestamps. The analysis revealed recognizable patterns among these values. That raised the question of IP address spoofing employment by the GFW. Delayed port scanning of probes IP addresses revealed that they generally belong to typical ISP clients.

In 2018, Dunna et al.[4] revisited past probing observations and suggested a simple but effective circumvention technique. They confirmed that probing is triggered by a Tor connection attempt and occurs on multiple ports, the port of Tor connection, and standard ports such as HTTP. Unlike previous studies, the filtering mechanism was found to block the entire IP address, but not an IP: port tuple. The blocking duration was reported to be about 12 hours, the same value as it was in earlier studies. Dunna et al. observed that most of the probing addresses appeared to be uniformly distributed and rarely reused. They suggested a simple strategy of not replying to probes as an effective way of mitigating probing attempts and preventing a Tor bridge from being blocked.

4.5 Tor pluggable transports

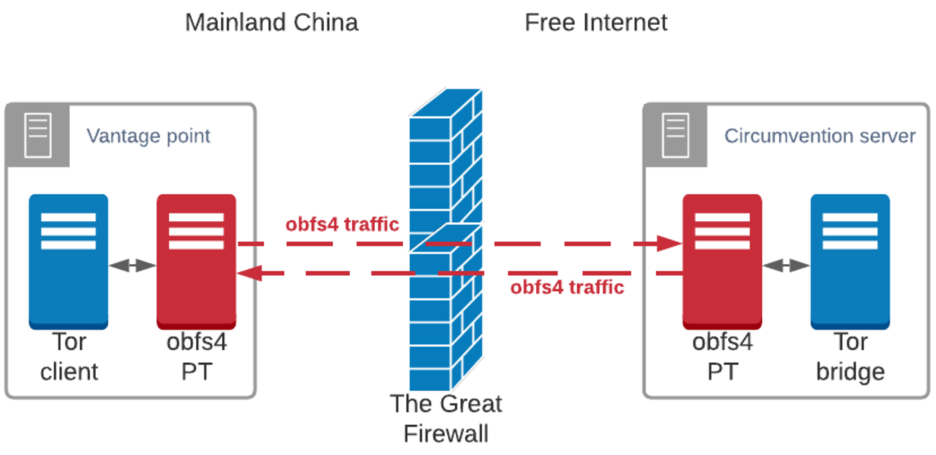


Figure 7: Typical setup for obfs4 pluggable transport.

In 2011, the arms race between the Chinese censor and Tor continued to expand.

As a result, Tor developers introduced support for pluggable transports (PT) to accelerate the development of prospective anti-censorship solutions [77, 78, .] Pluggable transports act as a SOCKS proxy for Tor endpoint. Furthermore, the Tor traffic flow between the bridge and the client is transformed by pluggable transports. The traffic flow transformation allows to bypass censor's detection and brings new anti-censorship capabilities not only for Tor but other circumvention solutions, such as Lantern and Psiphon. Figure 7 illustrates a typical PT setup.

Furthermore, the Tor project provided API and guidelines for potential PT developers; thus, dozens of transports have been developed and included in the Tor software bundle. However, many earlier versions of PT, such as obfs, obfs2, and obfs3, were deprecated as the firewall discovered ways to detect them. Currently, only obfs4 [79] and meek [6] are deployed in the Tor browser and recommended by Tor. Meek was developed by David Fifield and applied the domain fronting technique to obscure real destinations through third-party content delivery networks[6]. In contrast, obfs4 does not relay flows to third party infrastructure but obfuscates Tor traffic using Elligator elliptic-curve obfuscation [48] and ntor protocol [80] for one-way authentication. The result is randomly distributed traffic flow.

4.6 Summary

The chapter reveals the full architecture of the Tor anonymity network, which consist of: Onion Proxy, Tor directory authority, three different types of nodes, and bridges. It was also shown how Tor constructs circuits across the nodes, maintains IP-addresses anonymity, and facilitates censorship bypassing. The chapter demonstrates a fundamental failure in Tor's design, which leads to easy blocking of conventional Tor by preventing access to directory authority servers. Further, it was discussed how this flaw ends up in the arm-race between the Great Firewall and the Tor developers and pushed the development of bridges and pluggable transports. In addition, the entire section describes active probing, an ultimate reconnaissance mechanism that the Chinese censor actively uses against Tor infrastructure.

5 The Tor network reachability from mainland China

Censorship circumvention would be impossible without an understanding of blocking mechanics, which are employed by a censor. Therefore, this chapter provides details regarding how the firewall blocks all essential parts of the Tor network, including directory authority, guard relays, and bridges. The chapter aims to reveal what happens during blocking and revisits, confirms, and complements the results of previous observations[4, 3, 2, 55]. The goal is achieved by employing a series of connectivity tests describes in the next section.

5.1 Testing Methodology

For every test, the setup includes three to five servers in mainland China called vantage points. Chinese servers located in Tencent and Alibabcloud datacenters. Earlier tests were conducted using two Alibabacloud servers in Bejing and Shanghai. Three servers in Shanghai, Beijing, and Guangzhou from the Tencent data center were used as main testing servers. A total of five testing points provide necessary diversity in the server's network and physical locations. All vantage points were installed with Ubuntu 20.04 operation system with only one exception. Alibaba server in Beijing used Windows Server 2019.

The circumvention infrastructure, such as Tor bridges and relays, was deployed in the Amazon cloud server platform. Tor servers were placed in North Virginia, Paris, and Seoul to cover the wider geographical areas. Similar to Chinese servers, all circumvention servers worked under Ubuntu 20.04. Together with circumvention servers, a separate Tor bridge server was deployed in France in August 2020. This server was used to separate possible background scanning from the GFW and not participate in any tests. In addition, an iperf3 [81] server was deployed in London to facilitate bandwidth measurement.

The great advantage of cloud providers was an ability to change IP-addresses whenever it was necessary, thus for every experiment; all servers were designated with unique IP-addresses that were never reused. Furthermore, all firewalls were turn off to avoid any hypothetical connectivity disruption.

The deployed servers were used to conduct five experiments. The first three tests examined the availability of conventional Tor infrastructure. The first one examined the reachability of Tor directory authority. The second test evaluated the accessibility of Tor guard nodes. The third test demonstrated how fast the GFW blocks new Tor nodes. Two last tests were dedicated to Tor bridges. Test number four was about vanilla bridges. The fifth experiment was related to bridges with obfuscated pluggable transport support, i.e., obfs4[79]. All details relevant to a particular test setup are further described in corresponding sections.

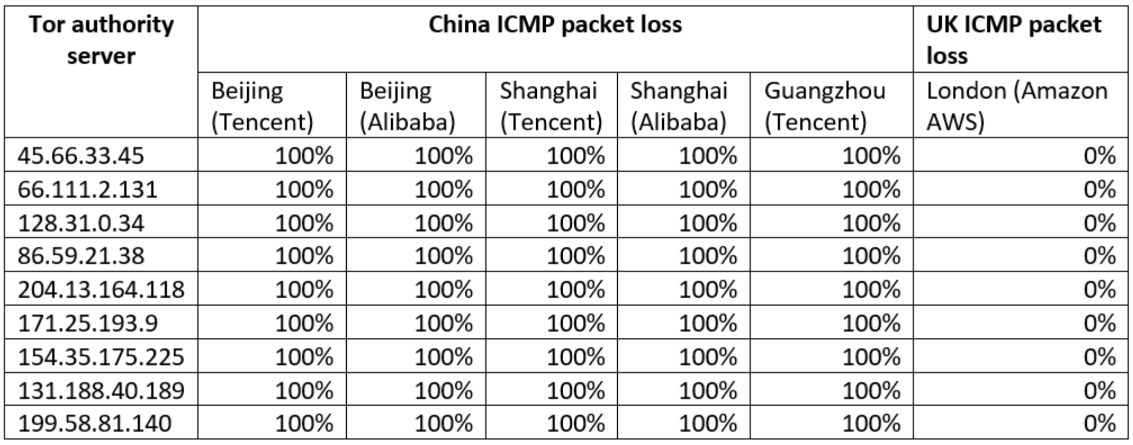
5.2 Availability of vanilla Tor infrastructure

This section is dedicated to the reachability of conventional Tor infrastructure, consisting of public relays and the directory authority. To the date of this writing (November 2020), the Tor network included about 6500 relays and ten official directory authority servers [82]. Although, only about 3500 of 6500 nodes are capable of handling Tor clients connections.

5.2.1 Reachability of Tor directory authority nodes

Initial onion routing proposals used in-band network status updates similar to conventional routing protocols [73]. However, Tor designers decided to discontinue this practice and instead implemented a centralized directory solution that relays on a limited number of directory nodes. Currently, this number is 10. Tor directory authority servers provide Tor network consensus information to all members. Moreover, for a Tor client, the Inability to fetch network consensus data results in a permanent network connectivity issue. It has been reported that Chinese censor exploits this flaw and permanently blocks access to directory servers, thus preventing vanilla Tor connectivity [2, 4].

The continuous ICMP ping and TCP SYN scanning were used to test the reachability of directory authority servers from within mainland China using all five vantage points. The test was conducted continuously for one hour and was performed three times in September, October, and November 2020. To show that directory authority servers were operational during the testing time, their availability was additionally checked from a separate control server in London. As the test showed, not a single ICMP or TCP SYN packet could reach authority servers from any vantage point for every attempt. This result indicates that connectivity is blocked on a per IP basis, probably with the BGP tampering technique. Table 1 illustrates detailed ICMP connectivity between vantage points and authority servers.

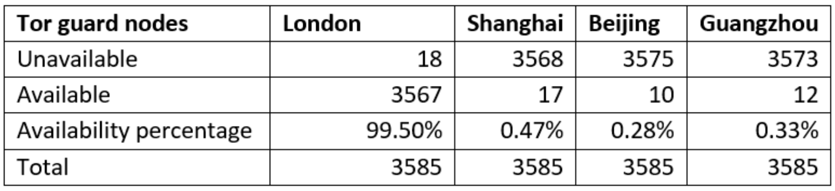


Tor authority server	China ICMP packet loss					UK ICMP packet loss
	Beijing (Tencent)	Beijing (Alibaba)	Shanghai (Tencent)	Shanghai (Alibaba)	Guangzhou (Tencent)	London (Amazon AWS)
45.66.33.45	100%	100%	100%	100%	100%	0%
66.111.2.131	100%	100%	100%	100%	100%	0%
128.31.0.34	100%	100%	100%	100%	100%	0%
86.59.21.38	100%	100%	100%	100%	100%	0%
204.13.164.118	100%	100%	100%	100%	100%	0%
171.25.193.9	100%	100%	100%	100%	100%	0%
154.35.175.225	100%	100%	100%	100%	100%	0%
131.188.40.189	100%	100%	100%	100%	100%	0%
199.58.81.140	100%	100%	100%	100%	100%	0%

Table 1: Availability of Tor authority nodes in mainland China.

5.2.2 Accessibility of Guard nodes

Ordinary nodes might relay the Tor network consensus information. Thus, authority servers blocking struggles to block the network entirely. Therefore, to complete the blocking target, the censor needs to block all entry points to the Tor network, i.e., all guard nodes. In order to test the availability of guard nodes, the complete list of 3585 guard nodes was acquired from the Tor metrics service via the onionoo protocol. Then the TCP SYN connection was sent to the published Tor port for every node. Every node connectivity was checked from three vantage points in China and the server in London to discard false-negative results if a measured node was not responding due to shutdown status or global connectivity issue. The experiment showed that most of the 3585 nodes are unavailable from China. Exhaustive values are presented in Table 2. This result corresponds well with observations from earlier studies[4, 3, 83]. Dunna et al.[4] demonstrated that only about 0.2% of Tor nodes are reachable. The conducted experiment showed a similar percent of reachable nodes, whereas most nodes are entirely blocked.



Tor guard nodes	London	Shanghai	Beijing	Guangzhou
Unavailable	18	3568	3575	3573
Available	3567	17	10	12
Availability percentage	99.50%	0.47%	0.28%	0.33%
Total	3585	3585	3585	3585

Table 2: Tor guard nodes availability.

5.2.3 Newly published relay blocking

A conventional Tor server was deployed in the Amazon AWS cloud to verify how long it takes for the GFW to block a newly published relay. To better understand the blocking mechanism, the connectivity between three vantage points in China and the new relay was checked every minute with ICMP requests continuously for 96 hours. As soon as the new server appeared in the public relay directory, it was blocked in 60 minutes and became unavailable from all vantage points. Previous studies showed that this interval used to be 10 minutes[4]. Apparently, the GFW enhanced the interval by now. Stopping the Tor relay, which deletes it from the public relay directory, did not immediately unblock the server. The server was blocked for more than 96 hours additionally. However, during this time, no active probing scanners were registered. Table 3 demonstrates the differences between the obtained results and previous observations from 2018[4], 2015 [3], and 2012 [2].

5.3 Availability of Tor Bridges

Bridges introduction was a reply to the increasing censorship of vanilla Tor. As it was mentioned in Chapter 4, bridges are hidden Tor nodes. To further increase their anti-censorship capabilities, Tor developers introduced pluggable transport support.

Comparison	Year			
	2012	2015	2018	2020
Block method	IP: port	IP: port	IP	IP
Block duration	12 hours	12 hours	12 hours	> 96 hours
Reaction time	no data	no data	10 minutes	60 minutes
Probing of a target server after a blocking event	no data	yes	yes	no

Table 3: Comparison of results from previous observations, in 2012 [2], in 2015[3], and 2018[4]

Most pluggable transports, such as obfs4 or meek, transform the traffic flow between a Tor client and a Tor bridge. Tor metrics showed that about 1800 bridges were operating in November 2020.

5.3.1 Vanilla Bridges

This test aimed to examine the firewall's reaction to the usage of unpublished Tor bridges. This test setup included an Alibaba cloud server in Shanghai and a Tor bridge in the Singapore region of Amazon AWS. The Tor client connectivity was imitated with the Tor Connection Initiation Simulator (TCIS) developed by Winter and Lindskog[2] and which was later used by Dunna et al. in their recent study of Tor bridges blocking[4]. The test was conducted in the same way as it was done in 2018 by Dunna et al., i.e., TCIS initiated the Tor connection to the server in Singapore, where potential probing traffic was captured for all ports.

Analysis of the captured traffic revealed no sights of scanning activity from the GFW. The bridge was accessible from China for 23 hours and was blocked after this time interval. This result showed that the firewall discontinued the vanilla Tor probing.

5.3.2 Bridges with obfs4 pluggable transport support

The setup for this experiment consists of three Tor bridges, i.e., circumvention servers, and three Tor clients, i.e., vantage points. Bridges were located in Amazon AWS datacenter in North American, European, and Asian regions. Tor clients were located in Tencent data center in Shanghai, Beijing, and Guangzhou. Each circumvention server was configured to act as a hidden obfuscated Tor bridge[84] with the support of obfs4 pluggable transport [79]. Each client was configured as an obfuscated Tor proxy with the support of obfs4. Additionally, a separate iperf3 server was configured to provide random traffic generation and utilize all available bandwidth completely. Each vantage point had an iperf3 client installed. All servers worked under Ubuntu 18.04 LTS, and all listening ports were randomly selected. Table 4 describes all server's characteristics.

The experiment was done in three simultaneous and independent pairs of endpoints. Shanghai vantage server corresponded to North American Tor bridge, Beijing vantage points were used together with European Tor bridge, and Guangzhou server was

Server	IP-address	Datacenter	Region	Software	Listening port
Circumvention servers	54.89.160.59	Amazon AWS	North America, Virginia	Vanilla Tor	7340
				obfs4 pluggable transport	27331
	15.188.8.200		Europe, Paris	Vanilla Tor	42058
				obfs4 pluggable transport	13889
	15.164.231.177		Asia, Seoul	Vanilla Tor	15324
				obfs4 pluggable transport	50688
Vantage points	42.192.58.168	Tencent	China, Shanghai	Vanilla Tor proxy, obfs4 transport, iperf3 client	No listening ports
	152.136.110.202		China, Beijing	Vanilla Tor proxy, obfs4 transport, iperf3 client	
	106.55.234.9		China, Guangzhou	Vanilla Tor proxy, obfs4 transport, iperf3 client	
Bandwidth test	52.47.136.152	Amazon AWS	Europe	iperf3 server	17568, 33026, 30244

Table 4: Experiment endpoints.

tightened with Asian bridge. Figure 8 depicts the experimental setup for a single pair of circumvention servers and vantage points.

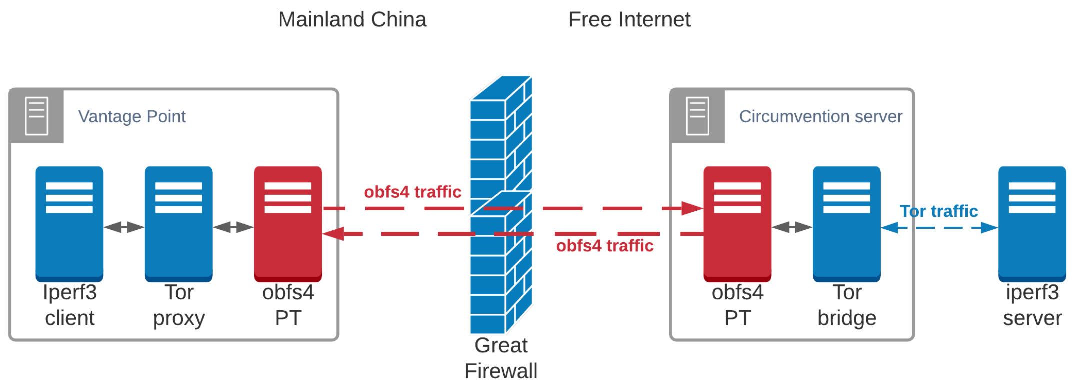
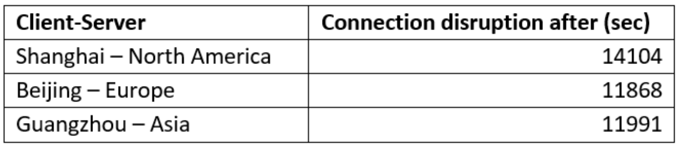


Figure 8: Experiment setup.

The following goals were established for the experiment. The first target was to show that the GFW can detect and disrupt randomly obfuscated circumvention protocols such as obfs4. The second aim was to find evidence of active probing usage against potential circumvention servers. The third goal was to collect and study the firewall's scanning traffic and its characteristics. In addition, the final objective was to Illuminate a potential trigger for active probing activity.

In order to achieve these objectives, the test began, simultaneously for all endpoints, with traffic capturing on both a vantage point and a circumvention server. All traffic coming from/to obfs4 port, which was selected randomly, was collected. Packets size, however, was truncated to 96 bytes to limit the volume of pcap files. Then, the iperf3 bandwidth test was launched in reverse mode, i.e., from server to client, via obfs4 proxy and Tor. The bandwidth test was stopped after 24 hours of execution. At the same time, Chinese servers were halted. The traffic capturing was continued for additional 24 hours in order to catch possible delay probes. Eventually, the experiment was halted after 48 hours from its beginning.

Connections between vantage points and corresponding circumvention servers were disrupted after about four hours of successful operation.



Client-Server	Connection disruption after (sec)
Shanghai – North America	14104
Beijing – Europe	11868
Guangzhou – Asia	11991

Table 5: Time for Tor obfs4 clients connectivity disruption.

In Figures 9-11, a rapid drop in packet rate indicates the moment of connectivity disruption. Any further communication attempts between Chinese clients and circumvention servers were constantly interrupted by the GFW. Iperf3 client continuous connections attempt provided a constant packets rate of about 20 packets per second.

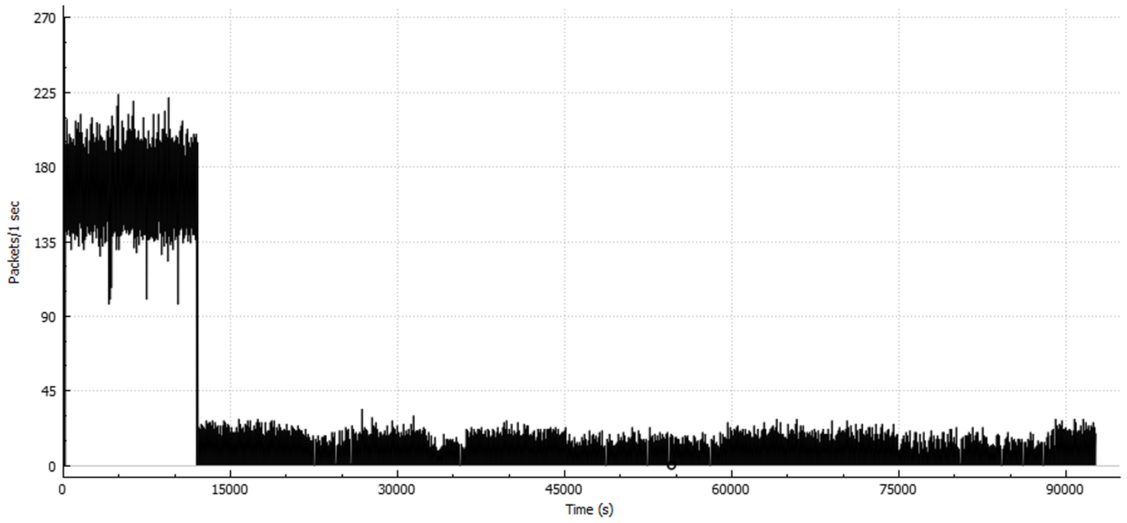


Figure 9: Beijing Tor obfs4 client drop in packet rate as a result of a connectivity disruption by the GFW.

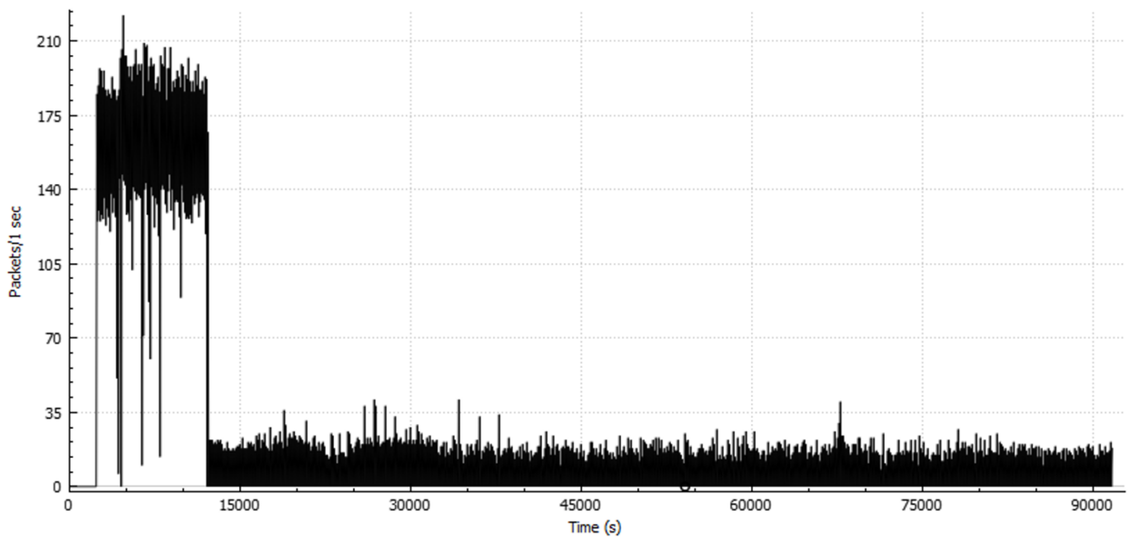


Figure 10: Guangzhou Tor obfs4 client drop in packet rate as a result of a connectivity disruption by the GFW.

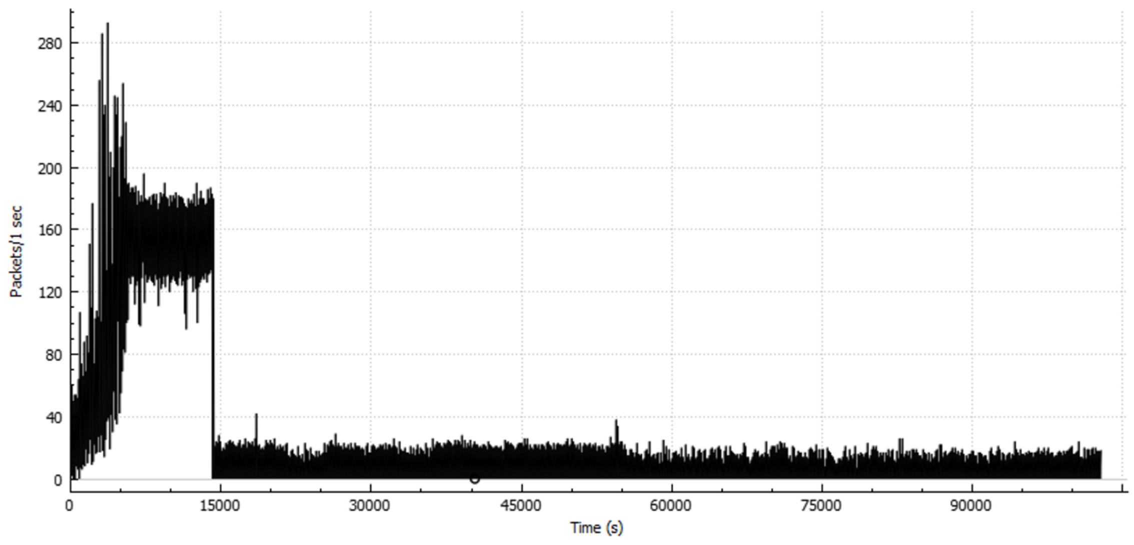
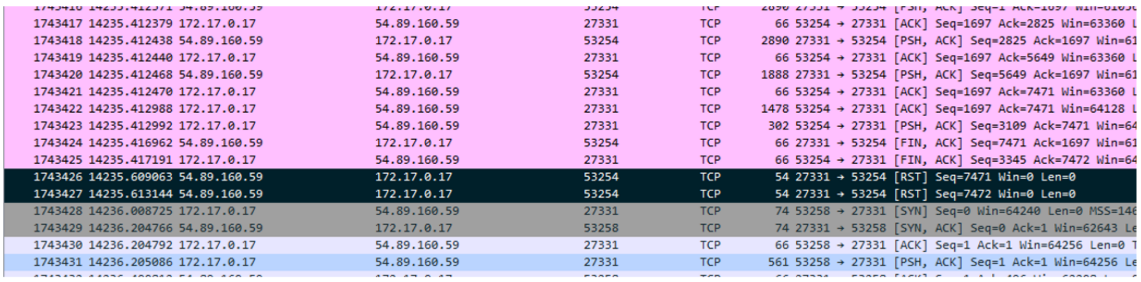


Figure 11: Shanghai Tor obfs4 client drop in packet rate as a result of a connectivity disruption by the GFW.

Figure 12 shows that the connectivity was disrupted with TCP reset packets. Any new connection attempts after the first interruption lead to connectivity disruption by means of TCP RST attack.

Further analysis showed that almost every new connection attempt from a client triggers probing activity. Moreover, the disruption of connectivity was correlated with the beginning of probing activity. This behavior was similar for all client-server pairs, as it is illustrated in Figure 14.

During the experiment, 960 unique probes were received by Tor bridges. American



```
1743417 14235.412379 172.17.0.17     54.89.160.59     27331    TCP     66 53254 → 27331 [ACK] Seq=1697 Ack=2825 Win=63360 L
1743418 14235.412438 54.89.160.59    172.17.0.17      53254    TCP   2890 27331 → 53254 [PSH, ACK] Seq=2825 Ack=1697 Win=61
1743419 14235.412440 172.17.0.17     54.89.160.59     27331    TCP     66 53254 → 27331 [ACK] Seq=1697 Ack=5649 Win=63360 L
1743420 14235.412468 54.89.160.59    172.17.0.17      53254    TCP   1888 27331 → 53254 [PSH, ACK] Seq=5649 Ack=1697 Win=61
1743421 14235.412470 172.17.0.17     54.89.160.59     27331    TCP     66 53254 → 27331 [ACK] Seq=1697 Ack=7471 Win=63360 L
1743422 14235.412988 172.17.0.17     54.89.160.59     27331    TCP   1478 53254 → 27331 [ACK] Seq=1697 Ack=7471 Win=64128 L
1743423 14235.412992 172.17.0.17     54.89.160.59     27331    TCP    302 53254 → 27331 [PSH, ACK] Seq=3109 Ack=7471 Win=64
1743424 14235.416962 54.89.160.59    172.17.0.17      53254    TCP     66 27331 → 53254 [FIN, ACK] Seq=7471 Ack=1697 Win=61
1743425 14235.417191 172.17.0.17     54.89.160.59     27331    TCP     66 53254 → 27331 [FIN, ACK] Seq=3345 Ack=7472 Win=64
1743426 14235.609063 54.89.160.59    172.17.0.17      53254    TCP     54 27331 → 53254 [RST] Seq=7471 Win=0 Len=0
1743427 14235.613144 54.89.160.59    172.17.0.17      53254    TCP     54 27331 → 53254 [RST] Seq=7472 Win=0 Len=0
1743428 14236.008725 172.17.0.17     54.89.160.59     27331    TCP     74 53258 → 27331 [SYN] Seq=0 Win=64240 Len=0 MSS=146
1743429 14236.204766 54.89.160.59    172.17.0.17      53258    TCP     74 27331 → 53258 [SYN, ACK] Seq=0 Ack=1 Win=62643 Le
1743430 14236.204792 172.17.0.17     54.89.160.59     27331    TCP     66 53258 → 27331 [ACK] Seq=1 Ack=1 Win=64256 Len=0 T
1743431 14236.205086 172.17.0.17     54.89.160.59     27331    TCP    561 53258 → 27331 [PSH, ACK] Seq=1 Ack=1 Win=64256 Le
```

Figure 12: Forged TCP reset packets used to disrupt connectivity.

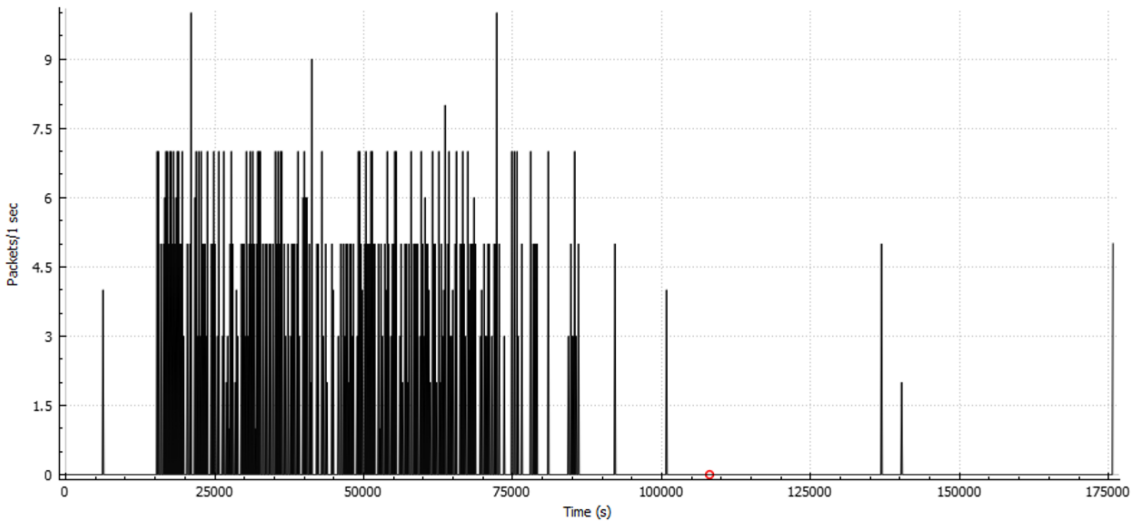


Figure 13: Probing activity against North America circumvention server logged for 48 hours.

circumvention server got 346 distinct probe sources. European server obtained 360 probe's addresses. Asian server got 254 probe's addresses. Chapter 6 provides a comprehensive analysis of received probes. Table 6 provide summary of the test result.

Client - Server	Connection disruption after (sec)	Number of probes	Probes arrived after (sec)
Shanghai – North America	14104	346	300
Beijing – Europe	11868	360	1200
Guangzhou – Asia	11991	254	1200

Table 6: Result summary.

5.4 Summary

All tests presented in this chapter were designed to clarify observations of the GFW conducted in previous studies [4, 3, 85, 2, 55, 83]. Tests confirmed that all Tor

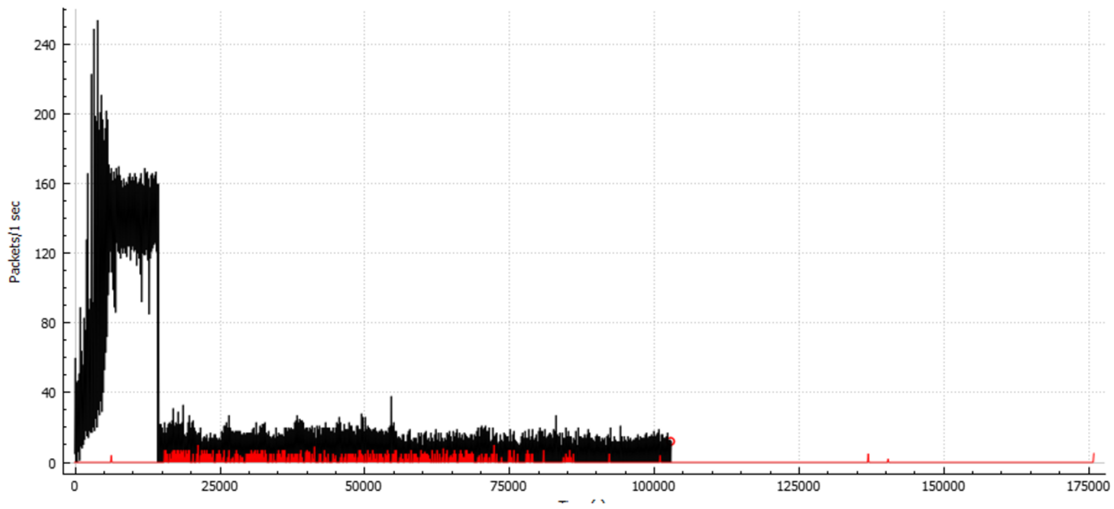


Figure 14: Correlation between Shanghai client's traffic received by North American circumvention server (black) and active probing packets engaged by the GFW (red).

Directory authority servers are blocked on a per IP base. It was verified that almost all guard nodes were also blocked with some minor exceptions. Furthermore, the blocking interval for newly published relays was found to be shifted from 10 minutes to 60 minutes. In contrast with earlier observations [4, 3, 2], the GFW continued to block a newly published relay for days after the relay shutdown and did not engage any active probing against the former relay. Another finding was the discontinue of legacy probing against vanilla Tor triggered by Tor TLS 1.0-1.2 handshake. The blocking method of vanilla Tor bridges has changed. Active probing started to perform a supplementary role and not the main trigger of the blocking mechanism as it was before.

Until recently, it was believed that obfs4 provides reliable protection again the GFW's censorship as the firewall cannot detect and block Tor bridges with obfs4 support [86]. Unfortunately, these tests proved that the GFW can detect and disrupt connections obfuscated with obfs4 transport and that the firewall engages active probing against obfs4 servers.

6 Active probing

Active probing empowers the censor to clarify the otherwise ambiguous traffic and established full control over it. Figure 4 illustrates the typical scheme of active probing which was introduced in 2010 as a subsystem of the GFW. Experiments four and five from Chapter 5 confirmed that the firewall still utilizes the active probing technique, almost a decade after its introduction. These tests collected significant data about recent firewall's scanning activities. Analysis of the collected data together with observations from earlier studies [4, 3, 2, 55] reveals the comprehensive picture of active probing and its evolution over the past years. Thus, this chapter highlights some previously unknown probing characteristics discovered during the previous chapter's tests and compares the legacy probing technique with recent observations.

6.1 Legacy probing

In 2011-2012, the probing was triggered by a unique list of ciphers used in the Tor TLS handshake [55, 2]. After a cipher list vulnerability was fixed, the Tor handshake's distinctive TLS characteristics were used by the firewall to detect Tor. All Tor version which used TLS 1.0 - 1.2 had unique handshake properties[47]. This issue was partially solved after the implementation of TLS 1.3 protocol support. However, even after the introduction of pluggable transports, the censor found a way to detect obfs2 and obfs3 with active probing[3].

Legacy probing was triggered by unique characteristics of the Tor TLS1.0-1.2 handshake. In 2012, the censor maintained a scanning queue, as probes arrived in 15 minutes intervals [2]. In 2015, Ensafi et al.[3] showed that the scanning queue was discontinued as probes started to arrive almost instantly after the Tor TLS negotiation. In 2012 and 2015, blocking was reported to base on IP: port tuple[2, 3]. In 2018, Dunna et al.[4] showed that the blocking mechanism had been changed, as the blocking was applied to the whole IP-address. Since 2015, probes were uniformly distributed with IP-addresses 202.108.181.70, and 111.202.242.93 were the most common sources of probes. The IP address 202.108.181.70 was disproportionately involved in active probing (sending about 50% of all probes)[3].

6.2 Nowadays probing

After the introduction of TLS1.3 in 2018, the handshake characteristics were not sufficiently distinct to trigger probing. Experiment 4 showed that the probing for TLS1.0 – 1.2 was discontinued as the probing was no triggered by Tor Connector Initiator Simulator by Winter and Linsock[2], which was used in previous studies [4, 3, 2]. TLS 1.3 handshake did not trigger scanning as well. Thus, either version of Tor TLS handshake does not rigger probing anymore.

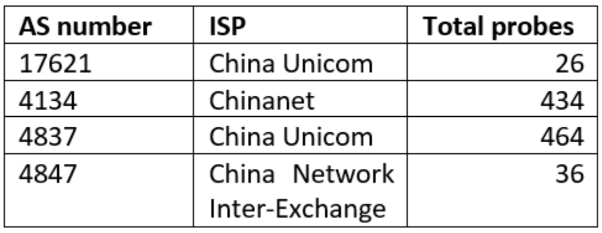
Furthermore, Experiment 5 detected 938 separate probe's addresses. Surprisingly, probing was used against a protocol with a random traffic flow which was design in such a way that its traffic is indistinguishable from an arbitrary sequence [79, 48]. This behavior raises the question about what activates the probing action?

It appears reasonable that entropy (randomness) of the traffic flow triggered the probing. Presumably, the bandwidth utilization had some influence as well. This conclusion seems reasonable as it is not common to encounter a randomly-distributed traffic flow on the Internet.

Tests confirmed that the blocking applied to a complete IP address and not to IP: port pair as it was in [3, 2]. No probing queues were detected; after the connection was disrupted, probes arrived almost instantly after the new connectivity attempt. In addition, the probe's sources were uniformly distributed.

6.2.1 Analysis of probing sources

Test 4 attracted thousands of unique probes from 960 IP-addresses. Analysis of probing sources revealed 916 IP addresses that were used only once and 22 IP addresses that were employed twice. This result corresponds well with earlier studies that showed that probes IP-addresses are rarely reused[4, 3]. In addition, no IP-address correlation was registered between 1201 probe's IP-addresses collected by Dunna et al. and 960 probe's addresses collected in this study. However, similarities between probes originating subnets were discovered. Table 7 demostrates the distribution of probes among Autonomous Systems. Figure 15 show the geographical distribution of the probe's sources.



AS number	ISP	Total probes
17621	China Unicom	26
4134	Chinanet	434
4837	China Unicom	464
4847	China Network Inter-Exchange	36

Table 7: Probes distribution among Autonomous Systems.

Nmap scanning of some probes IP addresses revealed regular ISP customers behind these addresses. The same result was snown in [3]. The previous most common probe's IP-addresses 202.108.181.70 and 111.202.242.93 were not detected. Figure 16 reveals how often probes sources IP addresses are reused. All these indicate the usage of temporary IP-address spoofing. The GFW shares active probing address space with ordinary users and temporary leases addresses for the purposes of active probing. The fact that no probes coming from sources outside China were ever detected additionally supports this claim.

Table 8 summarizes probing studies from different years and compares them with current observations.

6.2.2 Observing a single probe

A probe connection begins with a regular 3-way handshake following by a single payload packet. After receiving the payload ACK, the probe party closes the TCP connection using routine FIN, ACK sequence.

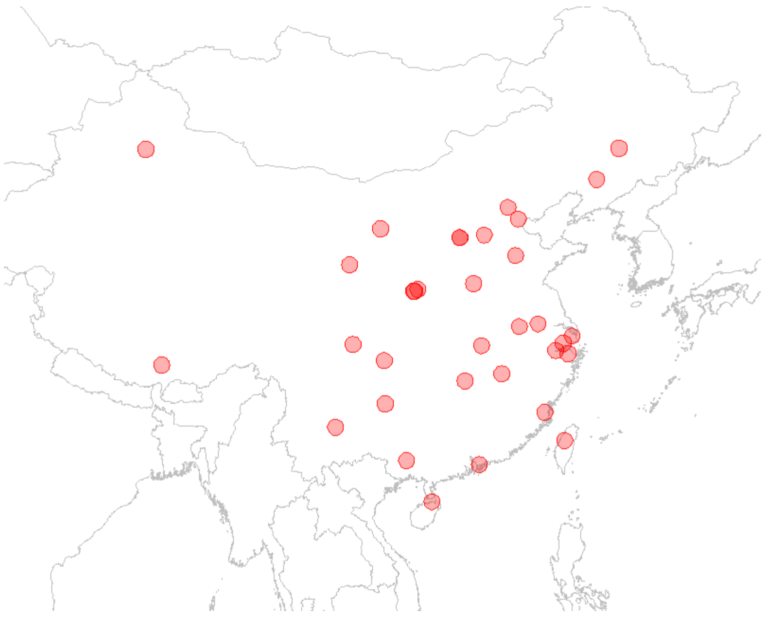


Figure 15: Probe's sources geolocations.

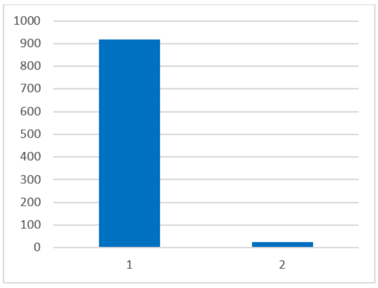
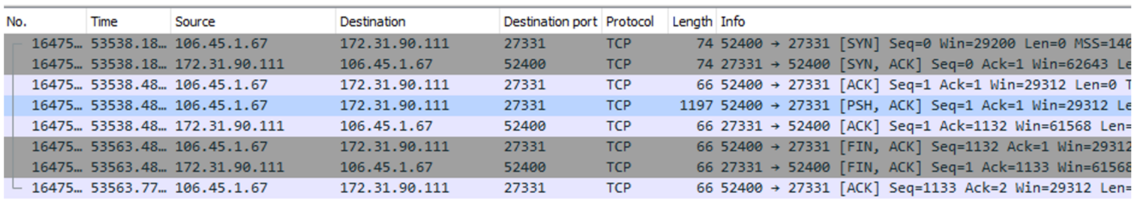


Figure 16: Only 22 out of 960 IP-addresses were used more than two times as probe sources.

Test 5 revealed that occasionally probe payloads contain replays of traffic previously sent by vantage points. For example, the payload showed in Figure 18 was received twice. This behavior indicates that the GFW uses previous packets replay as an active probing payload. Conversely, this is not always the case. Sometimes probing payloads look entirely random. As the payload was randomly generated, no coincidence possible.

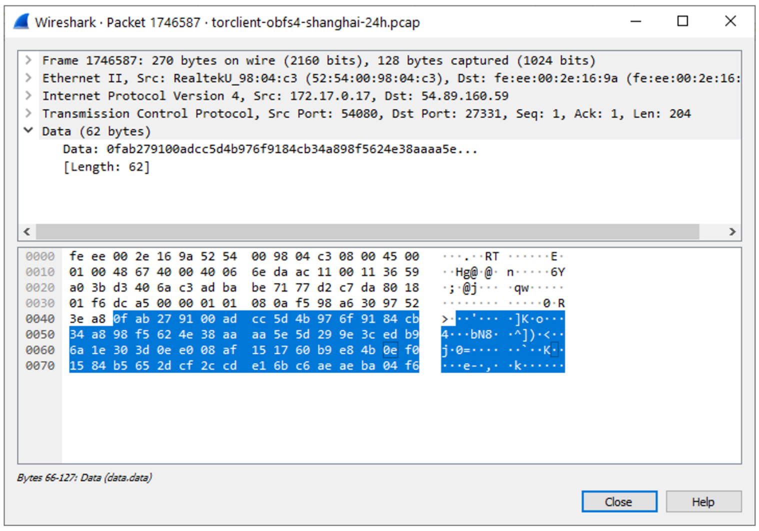


No.	Time	Source	Destination	Destination port	Protocol	Length	Info
16475…	53538.18…	106.45.1.67	172.31.90.111	27331	TCP	74	52400 → 27331 [SYN] Seq=0 Win=29200 Len=0 MSS=146
16475…	53538.18…	172.31.90.111	106.45.1.67	52400	TCP	74	27331 → 52400 [SYN, ACK] Seq=0 Ack=1 Win=62643 Le
16475…	53538.48…	106.45.1.67	172.31.90.111	27331	TCP	66	52400 → 27331 [ACK] Seq=1 Ack=1 Win=29312 Len=0 T
16475…	53538.48…	106.45.1.67	172.31.90.111	27331	TCP	1197	52400 → 27331 [PSH, ACK] Seq=1 Ack=1 Win=29312 Le
16475…	53538.48…	172.31.90.111	106.45.1.67	52400	TCP	66	27331 → 52400 [ACK] Seq=1 Ack=1132 Win=61568 Len=
16475…	53563.48…	106.45.1.67	172.31.90.111	27331	TCP	66	52400 → 27331 [FIN, ACK] Seq=1132 Ack=1 Win=29312
16475…	53563.48…	172.31.90.111	106.45.1.67	52400	TCP	66	27331 → 52400 [FIN, ACK] Seq=1 Ack=1133 Win=61568
16475…	53563.77…	106.45.1.67	172.31.90.111	27331	TCP	66	52400 → 27331 [ACK] Seq=1133 Ack=2 Win=29312 Len=

Figure 17: A probe connection in Wireshark.

Comparison	Year			
	2012	2015	2018	2020
Probe's trigger	Tor TLS handshake	Tor TLS handshake	Tor TLS handshake	Traffic pattern/randomnes
Tor nodes blocking duration	12 hours	12 hours	12 hours	More than 96 hours
Probing queue	15 minutes	Instantly	Instantly	Instantly
Probing distribution	Not Uniform	Uniform	Uniform	Uniform
Most common probe's source	202.108.181.70	202.108.181.70	111.202.242.93	-
Probes AS distribution	4134, 4837, 17622	4134, 4837, 7497	4134, 4837, 17676	4134, 4837, 4847, 17621

Table 8: Comparison of results from active probing studies in 2012 [2], 2015 [3], 2018 [4], and now 2020.


Wireshark · Packet 1746587 · torclient-obfs4-shanghai-24h.pcap

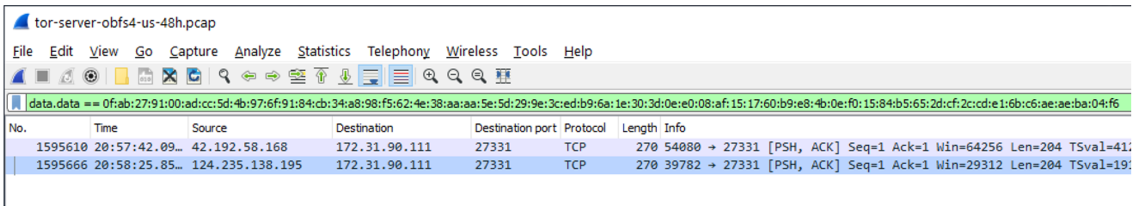
```
> Frame 1746587: 270 bytes on wire (2160 bits), 128 bytes captured (1024 bits)
> Ethernet II, Src: RealtekU_98:04:c3 (52:54:00:98:04:c3), Dst: fe:ee:00:2e:16:9a (fe:ee:00:2e:16:
> Internet Protocol Version 4, Src: 172.17.0.17, Dst: 54.89.160.59
> Transmission Control Protocol, Src Port: 54080, Dst Port: 27331, Seq: 1, Ack: 1, Len: 204
v Data (62 bytes)
     Data: 0fab279100adcc5d4b976f9184cb34a898f5624e38aaaa5e...
     [Length: 62]

0000  fe ee 00 2e 16 9a 52 54  00 98 04 c3 08 00 45 00   ···.··RT ······E·
0010  01 00 48 67 40 00 40 06  6e da ac 11 00 11 36 59   ··Hg@·@· n·····6Y
0020  a0 3b d3 40 6a c3 ad ba  be 71 77 d2 c7 da 80 18   ·;·@j··· ·qw·····
0030  01 f6 dc a5 00 00 01 01  08 0a f5 98 a6 30 97 52   ········ ·····0·R
0040  3e a8 0f ab 27 91 00 ad  cc 5d 4b 97 6f 91 84 cb   >···'··· ·]K·o···
0050  34 a8 98 f5 62 4e 38 aa  aa 5e 5d 29 9e 3c ed b9   4···bN8· ·^])·<··
0060  6a 1e 30 3d 0e e0 08 af  15 17 60 b9 e8 4b 0e f0   j·0=···· ··`··K··
0070  15 84 b5 65 2d cf 2c cd  e1 6b c6 ae ae ba 04 f6   ···e-·,· ·k······
```

Bytes 66-127: Data (data.data)

Figure 18: Payload that was sent from Shanghai vantage point at 20:57:41.

6.3 Mitigating active probing

When the probing was discovered for the first time, it was sufficient to change the list of ciphers in the Tor TLS handshake to mitigate it. Unfortunately, the censor rapidly adapted to this change. Moreover, the censor maintained constant monitoring and employed new probing techniques in reply to the introduction of new versions of


tor-server-obfs4-us-48h.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
data.data == 0f:ab:27:91:00:ad:cc:5d:4b:97:6f:91:84:cb:34:a8:98:f5:62:4e:38:aa:aa:5e:5d:29:9e:3c:ed:b9:6a:1e:30:3d:0e:e0:08:af:15:17:60:b9:e8:4b:0e:f0:15:84:b5:65:2d:cf:2c:cd:e1:6b:c6:ae:ae:ba:04:f6

No.      Time          Source           Destination    Destination port Protocol Length Info
 1595610 20:57:42.09… 42.192.58.168    172.31.90.111  27331            TCP         270 54080 → 27331 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=204 TSval=41
 1595666 20:58:25.85… 124.235.138.195  172.31.90.111  27331            TCP         270 39782 → 27331 [PSH, ACK] Seq=1 Ack=1 Win=29312 Len=204 TSval=19
```

Figure 19: Example of replay attack used by the firewall.

Tor and obfuscating protocols [78]. As a result, the constant evolution of probing significantly complicates the task of the probe's identification. Since different types of probes are used for different software versions and protocols, detecting every single probe's type is a tedious task. Although some earlier solutions tried to apply this strategy to mitigate probing [55]. Later, two distinct approaches for probing mitigating were used in different solutions. The first method is based on the filtering of all probing requests. The second approach is based on the disruption of probing triggering.

6.3.1 Filtering

The firewall decides to engage blocking based on the reply to the probing connection. This simple fact makes it possible to avoid blocking by not replying to a probing. However, it is necessary to identify either probing sources or the client's address in order to discard all probing traffic. Since probing identification is tedious and unreliable, client identification is the only option. Current solutions, such as obfs4 and Shadowsocks, identify clients by distributing out-of-band delivered secret which is used simultaneously for traffic encryption and probing identification. If the received payload is not encrypted by a previously agreed secret it is filtered and treated as an adversary action. Another method to identify legitimate clients utilizes a separate signaling channel to inform the server-side about valid client's addresses. Chapter 7 describes a solution that uses port knocking and Software Define Networks for this purpose.

6.3.2 Disruption of probing triggering

Previously it was enough to spot and alter the traffic part, which triggers active probing activity to mitigate it. Multiply changes were done in Tor TLS handshake and obfuscation protocols to stop the scanner. Today, probing is engaged against the random connections, which means randomness is a pattern that is evaluated by the DPI engine of the GFW. Thus, changes in this pattern should thwart the detection. Unfortunately, every new pattern can be recognized through machine learning. Thus, constant polymorphism of traffic patterns is required to cheat the firewall's machine learning engine. In what shape and how the circumvention connection traffic should be altered is yet to be discovered. It is a good topic for future studies. Most likely some AI capabilities should be utilized for unpredictable circumvention traffic shaping. To further speculate that could potentially lead to a battle of AI between a censor and circumvention solution developers.

6.4 Summary

This chapter describes the characteristics of probing traffic captured during Chapter 5 tests. It reveals probe sources geolocation and their distribution among Autonomous Systems. Additionally, the chapter compares the results of active probing studies in 2012, 2015, and 2018 with current observations and reveals the probing evolution.

Current observations confirm that probes IP-addresses are rarely reused and validate the IP-address spoofing for probes address space.

Section 6.2 reveals that nowadays, the TLS protocol handshake does not trigger the scanning and that the scanning is triggered, presumably, by traffic randomness. However, this hypothesis should find its confirmation in future studies. Section 6.2.2 revealed that sporadically probe payloads contain replays of traffic previously sent by legitimate clients. This behavior proves that the active probing utilizes a replay attack in the hope of improving circumvention software detectability. Thus, probes classification looks like a good topic for future studies.

Finally, section 6.2 introduces two common tactics of probing mitigating. The first tactic implies filtering or not responding to probing requests. This tactic is employed for the solution presented in this study. The second tactic suggests the disruption of probing triggering by protocol shaping in order to thwart probing triggering.

7 Bypassing the GFW

Every censorship circumvention system must deal with an advisory, which employs passive and active analysis against selected circumvention strategy. With dynamic analysis, a censor imitates legitimate clients to ensure the circumvention purposes of the system. The GFW active probing against Tor bridges is a brilliant example of this tactic. A circumvention solution that can recognized adversary actions among legitimate users will benefit in performance and mitigating capabilities. Thus, this chapter describes a solution to thwart the firewall's active scanning, thus avoiding blocking in case of ambiguous traffic inspection. Upcoming sections present the threat model, suggested solution, and its effectiveness evaluation. Additionally, the chapter clarifies the selection of software-defined networks (SDN) and port knocking as a signaling channel for the suggested solution.

7.1 Threat model

The threat model contains five main components: the firewall (the censor), the censored client, the circumvention server, the signaling channel, and censored destinations. Mitigation is successful when the client connects to the circumvention server, as the server provides access to the censored destinations. The client and the circumvention server cooperate with each other; in particular, the client shares its IP-address with the server via a signaling channel.

The firewall supervises a national network as wells as all traffic exchange points with the global internet. The firewall can monitor all connections and traffic flows across the national network and discard or permit any packets. Moreover, the censor can actively interact with any communication endpoint, i.e., injects and replay packets. The censored client resides within the network behind the firewall, while the circumvention server and censored destinations locate beyond the censor's authority. The firewall prevents direct communications between the censored client and destinations but allows connections to the circumvention server and the signaling channel. All firewalls' reconnaissance arsenal, such as traffic inspection and active probing, might be engaged against circumvention and signaling connections.

The censor does not control the circumvention server, the client, the signaling channel, and destinations. It cannot perform the man-in-the-middle attack due to the encrypted nature of communication between the circumvention server and the client. However, the signaling channel is susceptible to the man-in-the-middle attack. The client has an out-of-band channel which it can use to obtain circumvention software, port-knocking sequence, and cryptographic keys.

7.2 Mitigating solution

Mitigation of active scanning requires no actions from a circumvention server in reply to the firewall's probing requests. As a rule, such behavior hardcoded by the developers of mitigating solutions, e.g., obfs4[79], Shadowsocks[8], and MTproto[7]. However, design flaws and implementation vulnerabilities often leave probing attacks opportunities, as it was with earlier versions of Shadowsocks[87] or predecessors of obfs4[3]. Since most mitigating solutions operate custom application layer protocols, probing protection is limited to their own sockets. The GFW, however, engages scanning against multiply sockets and utilizes different probing types simultaneously[3, 4, 87].

In contrast, traffic engineering can deliver probing resistance to the whole circumvention server. SDN makes it possible to resist scanning attempts by discarding all probe's IP-addresses or exclusively permitting the client's addresses. Since the firewall rarely reuses probing IP addresses, probe's IP-addresses discarding is challenging to implement. Fortunately, it is possible to permit clients' IP addresses if finding a reliable way to notify a mitigating solution about IP-addresses used by clients.

Unfortunately, client-side IP-addresses tend to change dynamically, especially nowadays when end-user devices connect to different network types during a day. Therefore, the mitigating solution should be informed about the client's IP-addresses in case of a new connection attempt or during an event of the client's IP-address change.

Luckily, every IP packet contains addressing information regardless of higher layers protocols. Thus, the port knocking technique[88] can be used for clients' address ascertainment the same way it is used for port opening purposes. As a precaution and for better undetectability, the port knocking signaling should be done over a separate communication channel with the destination address differs from the IP-address of the circumvention server. Thus, several payload-less packets of the port knocking sequence that are occasionally sent via an independent channel are complicated to spot and tight with a circumvention solution.

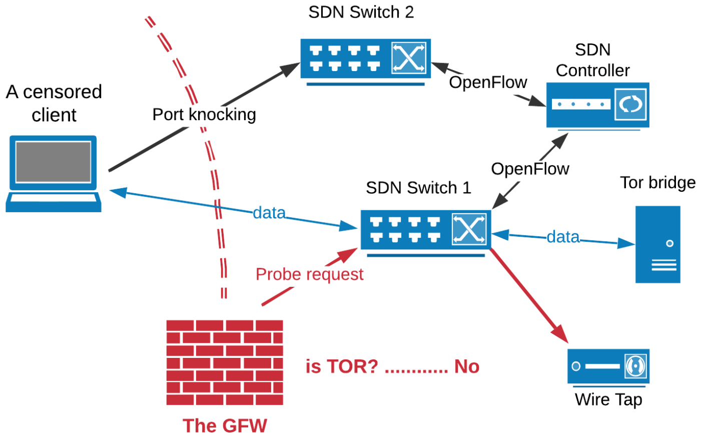


Figure 20: SND and port knocking setup for active probing circumvention.

Practical implementation of the suggested logic requires an SDN controller and two SDN switches. The first switch acts as a firewall gateway to the circumvention server, while the second switch waits for port-knocking signaling traffic from potential clients. The SDN controller manages flow entries for SDN switches, allows access to the circumvention server behind switch 1 for legitimate clients, and discards any scanning (probing) attempts from the GFW. Figure 20 depicts the principal solution scheme. The proposed SDN setup implements the following logic:

1. By default, any data plane traffic that comes to SDN Switch 1 is discarded regardless of the originating party; thus, limiting the possibility for the GFW to discover the Tor bridge with possible background scanning.

2. To connect to the Tor bridge, the censored client should send a port-knocking sequence to SDN Switch 2.

3. For SDN switch 2, the SDN controller keeps a history of incoming UDP packets and maintains a mealy machine.

4. If the censored client sends the valid packet sequence within a reasonable time interval to SDN switch 2, the SDN controller inserts flow entries that instruct SDNSwitch 1 to redirect future censored clients' traffic to the Tor bridge.

5. Starting from this point, the censored client might use the first SDN switch as a gateway to the Tor bridge.

7.3 Practical implementation

The proposed SDN setup was deployed in the Amazon AWS datacenter using two AWS EC2 instances installed with Open vSwitch[89] and a separate instance dedicated for Ryu SDN controller[90]. An additional cloud server was configured as a Tor bridge located in the stub network using Open vSwitch as a gateway.For the SDN controller, the UDP port knocking logic was implemented. The controller keeps the history of incoming UDP packets and maintains the Mealy machine depicted in figure 21.

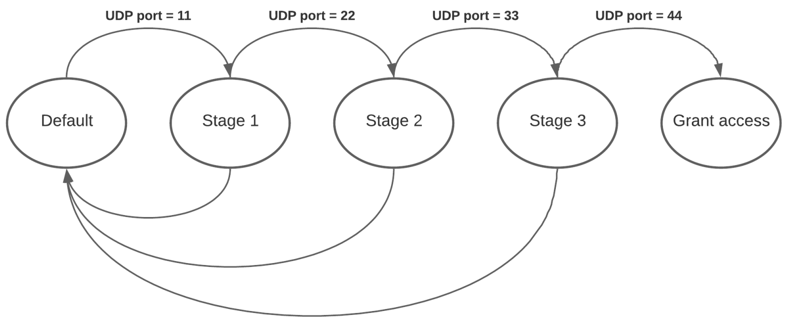


Figure 21: The Mealy state machine for current port knocking implementation with Ryu SDN controller

The state machine works as follows. All knocking packets should arrive in a reasonable time interval (20 seconds with the current code) from the same IP-address and in a predefined sequence(11,22,33,44). The first packet (UDP on port 11) initiates the port knocking sequence and changes the Mealy machine state to stage 1. The upcoming packet (UDP port 22) should arrive in 20 seconds; otherwise, the machine state goes back to default. The third and fourth packets follow the same logic. Eventually, once the last state is reached, the necessary action is performed.

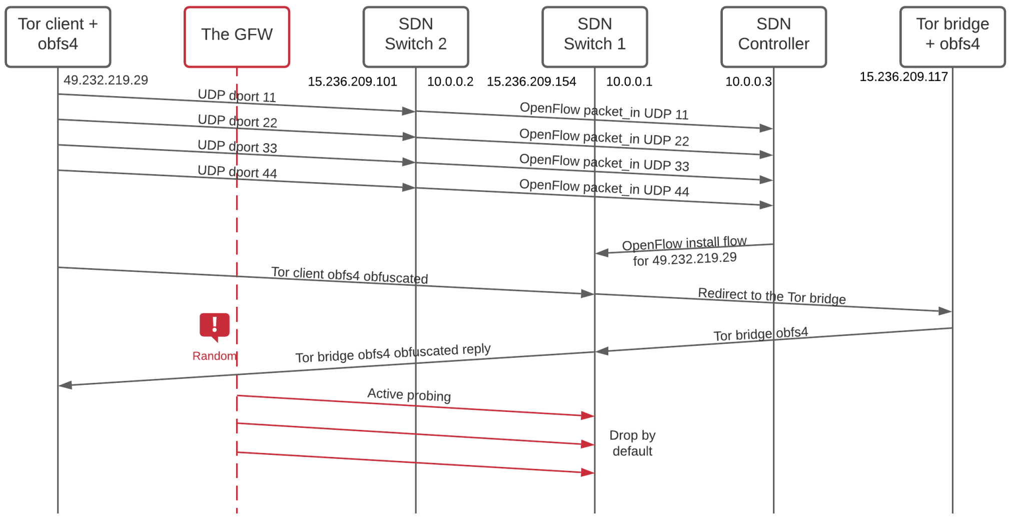


Figure 22: The traffic flow during the testing of the solution

Figure 22 shows the traffic flows between all components of the solution in practical implementation. First, the Tor client sent the port knocking packets sequence to the SDN Switch 2. The sequence is easily predictable but works for testing purposes because the GFW does not care much about tiny UDP payloadless packets. Linux nc command was used to send the knocking sequence as follows.

nc -u 15.236.209.101 11
nc -u 15.236.209.101 22
nc -u 15.236.209.101 33
nc -u 15.236.209.101 44

After the last packet arrived, the SDN controller installed a new flow to the SDN Switch 1. The new flow entry instructs the SDN Switch 1 to redirect all traffic coming from the Tor client's IP (49.232.219.29) to the Tor bridge obfs4 port and vice versa. Thus, the Tor client might select an arbitrary port to connect.

The new flow entry has an idle timeout of 3600 seconds. If the Tor client does not send any traffic in one hour, it should repeat the knocking sequence. The same is to apply when the client's IP-address has been changed. Since the flow entry had been installed, the client started to send traffic to SDN Switch 1, and the traffic was redirected to the Tor bridge. From this point, the firewall noted the substantial arbitrary traffic flow and started scanning against SDN Switch 1. By default, SDN Switch 1 dropped all firewall probes; thus, left the firewall in an uncertain state.

7.4 Effectiveness evaluation

The iperf3 bandwidth test was executed to assess the solution effectiveness in the same manner as in section 5.3.2. The test utilized all the available bandwidth and worked in reverse mode, which is natural for a production environment, i.e., the traffic flowed from the Tor bridge to the client. The test lasted for 86400 seconds (24 hours) and successfully delivered 2.38 Gbytes of data with an average speed of 236408 bits/sec. In contrast with the results obtained in Chapter 5, the iperf3 connectivity was not disrupted. The traffic flow experienced nor interruptions neither TCP reset attacks engagements. Figure 23 compares bandwidth testing data of the current test with the data obtained in Chapter 5. On the chart, the black line represents traffic flow for the solution testing, while the red line represents ipef3 data for raw obfs4 test of Beijing's client connectivity in section 5.3.2 of the fifth chapter, which is depicted by Figure 9.

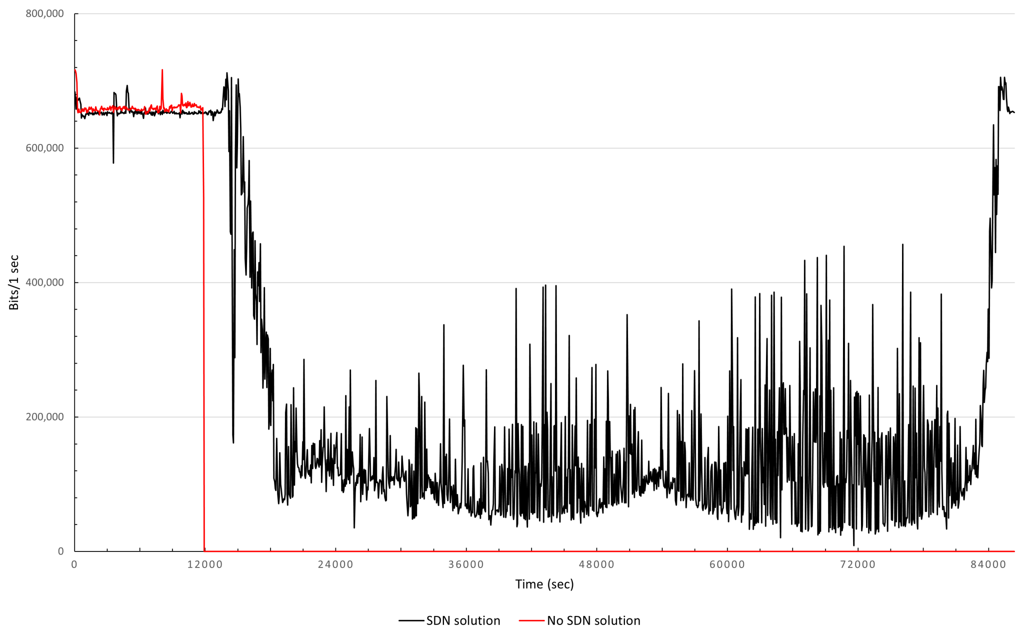


Figure 23: Iperf3 bandwidth comparison between the solution testing and iperf3 bandwidth for Beijing obfs4 connectivity test from section 5.3.2

Although the traffic flow was not blocked entirely, and the client retains its connectivity through the entire experiments, the test demonstrated the usage of throttling against the mitigating connection. Thus, the solution demonstrates its effectiveness against active probing, but the overall circumvention performance is limited due to the throttling used by the firewall to suppress the mass scale circumvention. It additionally supports the conclusion made in Chapter 5 that blocking is based on traffic characteristics and not an active probing engagement as it was previously.

7.5 Limitations

Although the solution demonstrates successful firewall penetration, several weaknesses should be fixed in future versions. The first flaw caused by the replay attacks vulnerability inherent to the traditional port knocking method[91]. With current realization, the same port numbers are used for every client. Therefore, the censor might bypass the scanning protection by intercepting and resending the knocking sequence. The port knocking sequence must be unique not for every client but for every particular signaling event in order to avoid possible replay attacks. The necessary replay attack resistance can be achieved by utilizing One-Time Password (OTP) with infinite nested hash chains[92] to generate as many unique knocking sequences as required. Another option is to employ the source port knocking method developed by Fakariah et .al[93].

This method twice reduces the number of knocking packets by utilizing source port information in the verification process. For every knock, the destination port and the source port become the meaningful fields. It does not compromise the security as the brute force complexity(C) remains the same, $C = 65535^4$.

The second limitation comes from the detectability of UDP packets and the possible blocking of the signaling channel by IP-address. Even if it is not an issue in testing deployment, the mass scale application requires a way to guarantee the resistance of the signaling channel. Such resistance might be achieved by increasing the number of terminating points and utilizing a domain fronting[6] for the knocking traffic. Domain fronting is expensive for mass-scale circumvention as the solution maintainers must pay for third party Content Delivery Network (CDN) infrastructure. However, CDN costs in case of a tiny amount of signaling traffic can be neglected.

The third problem is the bandwidth throttling of the data channel. Throttling is a common issue in GFW's circumvention. Such solutions as SSH tunnels and Shadowsocks struggle from throttling. Most likely, it happens once the defined traffic consumption threshold is reached, as, in this study, it occurred after about four hours of full bandwidth utilization. However, further investigation is required to understand the patterns behind throttling.

7.6 Summary

This chapter suggests the Great Firewall's mitigating strategy based on traffic engineering using SDN and port-knocking. It describes the threat model and practical implementation of the suggested strategy, including mealy state machine and SDN infrastructure details. As a consequence, the performance of the suggested solution was compared with reference tests from section 5.3.2. Upon comparison, the solution demonstrates successful circumvention and limited effectiveness. The effectiveness is limited due to the firewall's bandwidth throttling and the current implementation flaws, such as replay attack vulnerability and the signaling channel's detectability. Finally, some ways to improve current implementation were suggested.

8 Conclusion and discussion

In this work, numerous tests and observations were conducted to illuminate details of the arms race between the Tor anonymity network and the Great Firewall. It was shown that the firewall's active probing or scanning still contributes to the blocking decision. Furthermore, the study reveals that the scanning is not triggered by Tor handshake anymore but rather by the random nature of traffic, although this suggestion should be further investigated. Many previous probing characteristics were confirmed, and some new notable features were found. Finally, the mitigating solution was proposed based on software-defined networking and port knockin technologies. The solution proved its effectiveness despite being limited by bandwidth throttling. The list below enumerates all key findings and conclusions of the study.

- The Tor network was found to be still heavily suppressed in China. The tests showed that 99.5%of Tor's guard relays, including ten Tor directory authority nodes, are entirely blocked.
- Analysis of 960 scanning sources confirms that probe's IP-addresses are rarely reused[3]; only 22 out of 960 IP were detected twice. Further Nmap scanning confirmed that the firewall's probing engine shares IP-address space with ordinary users. It was also demonstrated that probing sources are evenly distributed across the country (Figure 15) and that AS distribution remains the same as in 2011.
- The comparison of modern scanning activity with probing observation form 2012, 2015, 2018 reveals the change in blocking duration for former Tor nodes from 12 hours[2] to more than 96 hours. The newly introduced Tor public relay was blocked in 60 minutes, and the server's IP remained blocked for 96 hours after the Tor service was shouted down. Previously it was 10 minutes and 12 hours[4]. This work shows that the GFW increased these timeouts.
- The pcap captures confirmed that the connectivity was disrupted with TCP reset attack using forged TCP reset packets in the same way as observed earlier[28].
- The solution testing showed that the firewall's active probing or scanning still contributes to the blocking decision. However, the test from section 5.3.1 showed that TCIS with traditional TLS 1.2 handshake does not trigger active probing anymore; supposedly, the traffic randomness is the trigger.
- It was demonstrated that the GFW successfully engages blocking against obfs4 transport and suppress obfs4 connections in 3-4 hours. However, it might be not only because of the obfs4 vulnerabilities but also because of obfs4 traffic randomness nature. This issue requires further investigation. The solution testing also demonstrated that 3-4 hours is the reaction time for actions against randomly obfuscated circumvention connection.

- Another key finding is the usage of user's traffic payloads in payloads of some probing packets. It indicates that the firewall's probing engine utilizes user's packets replay for scanning activities.

The work suggests two ways to thwart active probing, triggering avoidance and scanning request filtering. The last is used in the proposed mitigating solution. The solution demonstrates how traffic engineering and port knocking can thwart active probing and enhance connectivity for obfs4 at least for 24 hours. The solution effectiveness was evaluated against random obfs4 traffic flow and appeared to be sufficient to prevent the complete blocking of obfs4 connections. However, in the current implementation, the solution is limited with port knocking packets replay vulnerability and insecurity of the signaling channel. Fortunately, these flaws can be resolved with OTP port knocking[92] and domain fronting[6].

This study does not provide an answer to the third limitation, bandwidth throttling. The throttling was observed in all conducted tests and started after 3-4 hours of active (full bandwidth capacity) connection usage. Arguably, the throttling was engaged because of the high bandwidth utilization for several hours. However, there is no answer, whether all or only some traffic types invoke throttling and how the throttling is used in general by Chinese censor. This topic seems like a good opportunity for further studies.

Active probing triggering is another excellent topic, i.e., to what extent does traffic randomness (shape) trigger scanning. How active probing and bandwidth throttling are related to traffic patterns? Are there ways to alter the traffic shape to thwart these two censorship techniques? Two central questions for future studies.

References

[1] C. Sonali, J. Zang, Y. Yu, J. Sun, and Z. Zhang, "The golden shield project of China: A decade later-an in-depth study of the great firewall," *Proceedings - 2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, CyberC 2019*, pp. 111–119, 2019. DOI: 10.1109/CyberC.2019.00027.

[2] P. Winter and S. Lindskog, "How the great firewall of China is blocking tor," *2nd USENIX Workshop on Free and Open Communications on the Internet, FOCI 2012, co-located with USENIX Security 2012*, no. March, 2012.

[3] R. Ensafi, D. Fifield, P. Winter, N. Feamster, N. Weaver, and V. Paxson, "Examining how the great firewall discovers hidden circumvention servers," *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC*, vol. 2015-Octob, no. September, pp. 445–458, 2015. DOI: 10.1145/2815675.2815690.

[4] A. Dunna, C. O'Brien, and P. Gill, "Analyzing China's blocking of unpublished tor bridges," *8th USENIX Workshop on Free and Open Communications on the Internet, FOCI 2018, co-located with USENIX Security 2018*, no. Table 2, 2018.

[5] Kevin Bock, *Exposing and Circumventing China Censorship of ESNI*, 2020. [Online]. Available: https://geneva.cs.umd.edu/posts/china-censors-esni/esni/.

[6] D. Fifield, C. Lan, R. Hynes, P. Wegmann, and V. Paxson, "Blocking-resistant communication through domain fronting," *Proceedings on Privacy Enhancing Technologies*, vol. 2015, no. 2, pp. 46–64, 2015. DOI: 10.1515/popets-2015-0009.

[7] Telegram, *MTProto Mobile Protocol.* [Online]. Available: https://core.telegram.org/mtproto.

[8] Shadowsock Developers Team, "Shadowsocks : A secure SOCKS5 proxy," 2019.

[9] Getlantern.org, *Lantern Project, Lampshade: a transport between Lantern clients and proxies.* [Online]. Available: https://godoc.org/github.com/getlantern/lampshade.

[10] X. Zeng, X. Chen, G. Shao, T. He, Z. Han, Y. Wen, and Q. Wang, "Flow context and host behavior based shadowsocks's traffic identification," *IEEE Access*, vol. 7, pp. 41 017–41 032, 2019, ISSN: 21693536. DOI: 10.1109/ACCESS.2019.2907149.

[11] S. Frolov, J. Wampler, and E. Wustrow, "Detecting Probe-resistant Proxies," no. February, 2020. DOI: 10.14722/ndss.2020.23087.

[12] Fortune.com, *Fortune 500 2020*, 2020. [Online]. Available: https://fortune.com/fortune500/search/ (visited on 12/20/2020).

[13] Internet Society, "Internet Society Perspectives on Internet Content Blocking: An Overview," *Internet Society*, no. March, 2017.

[14] European Commission, "A Europe That Protects: Countering Illegal Content Online," Tech. Rep. October, 2019, pp. 3–5. [Online]. Available: https://ec.europa.eu/newsroom/dae/document.cfm?doc%7B%5C_%7Did=50096.

[15] UK Parliament, *Online harms Whitepaper*, April. 2019, ISBN: 9781528610803.

[16] US Congress, "SAVE Act," no. Qualitative Research Copyright © 2001 SAGE Publications (London, Thousand Oaks,CA and New Delhi) vol. 1(1): 23-46. [1468-7941 (200104) 1:1; 23-46; 016167] 23 QR, pp. 1–3, 2013.

[17] ——, *Children's Internet Protection Act*, 2000. [Online]. Available: https://www.congress.gov/bill/106th-congress/house-bill/4600?s=1%7B%5C&%7Dr=67.

[18] A. Richter, "Regulation of online content in the Russian Federation," 2015.

[19] EU Parliament, *DIRECTIVE (EU) 2017/541 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 15 March 2017 on combating terrorism and replacing Council Framework Decision 2002/475/JHA and amending Council Decision 2005/671/JHA*, 2017. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32017L0541.

[20] ——, *Directive 2011/93/EU of the European Parliament and of the Council of 13 December 2011 on combating the sexual abuse and sexual exploitation of children and child pornography, and replacing Council Framework Decision 2004/68/JHA*, 2011. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32011L0093.

[21] ——, *Directive (EU) 2019/790 of the European Parliament and of the Council of 17 April 2019 on copyright and related rights in the Digital Single Market*, 2019. [Online]. Available: https://eur-lex.europa.eu/eli/dir/2019/790/oj.

[22] Kingdom of the Netherlands, *Joint statement on the Directive of the European Parliament and Council on copyright in the Digital Single Market*, 2019. [Online]. Available: https://www.permanentrepresentations.nl/documents/policy-notes/2019/02/20/joint-statement-regarding-the-copyright-directive (visited on 12/20/2020).

[23] United Nations, *The Universal Declaration of Human Rights*, 1948. [Online]. Available: https://www.un.org/en/universal-declaration-human-rights/.

[24] Freedom House, "Freedom on the net," 2020, ISSN: 02624079. DOI: 10.1016/s0262-4079(15)60421-3. [Online]. Available: https://freedomhouse.org/sites/default/files/2020-10/10122020%7B%5C_%7DFOTN2020%7B%5C_%7DComplete%7B%5C_%7DReport%7B%5C_%7DFINAL.pdf.

[25] European Commission, *Open Internet*. [Online]. Available: https://ec.europa.eu/digital-single-market/en/policies/open-internet (visited on 12/20/2020).

[26] E. Claessen, "Reshaping the internet – the impact of the securitisation of internet infrastructure on approaches to internet governance: the case of Russia and the EU," *Journal of Cyber Policy*, vol. 5, no. 1, pp. 140–157, 2020, ISSN: 2373-8871. DOI: 10.1080/23738871.2020.1728356.

[27] M. Wander, C. Boelmann, L. Schwittmann, and T. Weis, "Measurement of globally visible DNS injection," *IEEE Access*, vol. 2, pp. 526–536, 2014, ISSN: 21693536. DOI: 10.1109/ACCESS.2014.2323299.

[28] N. Weaver, R. Sommer, and V. Paxson, "Detecting Forged TCP Reset Packets," *Proceedings of NDSS*, p. 15, 2009. [Online]. Available: papers://9934fd52-194f-4a2f-b130-d9de48bdab09/Paper/p69.

[29] Google, *HTTPS encryption on the web*, 2020. [Online]. Available: https://transparencyreport.google.com/https/overview?hl=en%7B%5C&%7Dtime%7B%5C_%7Dos%7B%5C_%7Dregion=chrome-usage:1;series:time;groupby:os%7B%5C&%7Dlu=load%7B%5C_%7Dos%7B%5C_%7Dregion%7B%5C&%7Dload%7B%5C_%7Dos%7B%5C_%7Dregion=chrome-usage:1;series:page-load;groupby:os (visited on 12/21/2020).

[30] R. S. Raman, "Censored Planet Observatory," Tech. Rep., 2020. [Online]. Available: https://ooni.org/documents/imv2020-slides/censored-planet.pdf.

[31] Internet Engineering Task Force (IETF), *RFC 6066 Transport Layer Security (TLS) Extensions: Extension Definitions*, 2011. [Online]. Available: https://tools.ietf.org/html/rfc6066 (visited on 12/21/2020).

[32] Z. Chai, A. Ghafari, and A. Houmansadr, "On the importance of encrypted-SNI (ESNI) to censorship circumvention," *9th USENIX Workshop on Free and Open Communications on the Internet, FOCI 2019, co-located with USENIX Security 2019*, 2019.

[33] R. Antonello, S. Fernandes, C. Kamienski, D. Sadok, J. Kelner, I. Gódor, G. Szabó, and T. Westholm, "Deep packet inspection tools and techniques in commodity platforms: Challenges and trends," *Journal of Network and Computer Applications*, vol. 35, no. 6, pp. 1863–1878, 2012, ISSN: 10848045. DOI: 10.1016/j.jnca.2012.07.010.

[34] Internet Engineering Task Force (IETF), *RFC 7858 Specification for DNS over Transport Layer Security (TLS)*, 2016. [Online]. Available: https://tools.ietf.org/html/rfc7858 (visited on 12/23/2020).

[35] IETF, *RFC 8484 DNS Queries over HTTPS (DoH)*, 2018. [Online]. Available: https://tools.ietf.org/html/rfc8484 (visited on 12/23/2020).

[36] P. Wu, *DNS Encryption Explained*, 2019. [Online]. Available: https://blog.cloudflare.com/dns-encryption-explained/ (visited on 12/23/2020).

[37] Internet Engineering Task Force (IETF), *RFC7230 Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing.* [Online]. Available: https://tools.ietf.org/html/rfc7230 (visited on 12/23/2020).

[38] IETF, *RFC7540 Hypertext Transfer Protocol Version 2 (HTTP/2)*, 2015. [Online]. Available: https://tools.ietf.org/html/rfc7540 (visited on 12/23/2020).

[39] Network Working Group, *RFC1928 SOCKS Protocol Version 5*, 1996. [Online]. Available: https://tools.ietf.org/html/rfc1928 (visited on 12/23/2020).

[40] OpenVPN, *Overview of OpenVPN*. [Online]. Available: https://community.openvpn.net/openvpn/wiki/OverviewOfOpenvpn (visited on 12/23/2020).

[41] J. J. K. Eric F Crist, *Mastering OpenVPN*. Packt Publishing Ltd, 2015, pp. 10–11, ISBN: 978-1-78355-313-6.

[42] OpenVPN, *OpenVPN 2.4 reference manual*, 2020. [Online]. Available: https://openvpn.net/community-resources/reference-manual-for-openvpn-2-4/ (visited on 12/23/2020).

[43] J. A. Donenfeld, "WireGuard: Next Generation Kernel Network Tunnel," pp. 1–20, 2017. DOI: 10.14722/ndss.2017.23160.

[44] Rohde & Schwarz GmbH & Co., *Rohde & Schwarz Adds Emerging WireGuard VPN Protocol to its Deep Packet Inspection (DPI) Software Library, R&S®PACE 2*, 2019. [Online]. Available: https://www.ipoque.com/news-media/press-releases/rohde-amp-schwarz-adds-emerging-wire-guard-vpn-protocol-to-its-deep-packet-inspection-dpi-software-library-r-amp-s-pace-2 (visited on 12/23/2020).

[45] OpenVPN, *Overview of changes in OpenVPN v2.4*, 2018. [Online]. Available: https://github.com/OpenVPN/openvpn/blob/release/2.4/Changes.rst (visited on 12/23/2020).

[46] Ssh.com, *SSH tunnel*, 2020. [Online]. Available: https://www.ssh.com/ssh/tunneling/ (visited on 12/23/2020).

[47] T. project, *Tor TLS history*. [Online]. Available: https://gitlab.torproject.org/legacy/trac/-/wikis/org/projects/Tor/TLSHistory0A (visited on 12/01/2020).

[48] D. J. Bernstein, M. Hamburg, A. Krasnova, and T. Lange, "Elligator: Elliptic-curve points indistinguishable from uniform random strings," *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 967–979, 2013, ISSN: 15437221. DOI: 10.1145/2508859.2516734.

[49] S. Khattak, M. Javed, P. D. Anderson, and V. Paxson, "Towards illuminating a censorship monitor's model to facilitate evasion," *3rd USENIX Workshop on Free and Open Communications on the Internet, FOCI 2013, co-located with USENIX Security 2013*, pp. 1–7, 2013.

[50] Y. Xu, *Deconstructing the Great Firewall of China*, 2016. [Online]. Available: https://blog.thousandeyes.com/deconstructing-great-firewall-china/ (visited on 10/07/2020).

[51] W. Banday, *The Great Firewall of China: A Digital Black Hole*, 2019. [Online]. Available: https://blog.catchpoint.com/2019/11/18/great-firewall-china-dns-ip-censorship/ (visited on 10/07/2020).

[52] Anonymous, "Towards a comprehensive picture of the Great Firewall's DNS censorship," *4th USENIX Workshop on Free and Open Communications on the Internet, FOCI 2014, co-located with USENIX Security 2014*, 2014.

[53] Sparks, "The collateral damage of internet censorship by DNS injection," *Computer Communication Review*, vol. 42, no. 3, pp. 22–27, 2012, ISSN: 01464833. DOI: 10.1145/2317307.2317311.

[54] A. A. Niaki, N. P. Hoang, P. Gill, and A. Houmansadr, "Triplet censors: Demystifying great Firewall's DNS censorship behavior," *FOCI 2020 - 10th USENIX Workshop on Free and Open Communications on the Internet, co-located with USENIX Security 2020*, 2020.

[55] T. Wilde, *Great Firewall Tor Probing*, 2011. [Online]. Available: https://gist.github.com/twilde/da3c7a9af01d74cd7de7%7B%5C#%7Dfile-00-report-txt (visited on 10/21/2020).

[56] C. Arthur, *China tightens 'Great Firewall' internet control with new technology*, 2012. [Online]. Available: https://www.theguardian.com/technology/2012/dec/14/china-tightens-great-firewall-internet-control (visited on 11/12/2020).

[57] J. Cheng, Y. Li, C. Huang, A. Yu, and T. Zhang, "ACER: detecting Shadowsocks server based on active probe technology," *Journal of Computer Virology and Hacking Techniques*, vol. 16, no. 3, pp. 217–227, 2020, ISSN: 22638733. DOI: 10.1007/s11416-020-00353-z. [Online]. Available: https://doi.org/10.1007/s11416-020-00353-z.

[58] Z. Wang, Y. Cao, Z. Qian, C. Song, and S. V. Krishnamurthy, "Your state is not mine: A closer look at evading stateful internet censorship," *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC*, vol. Part F1319, pp. 114–127, 2017. DOI: 10.1145/3131365.3131374.

[59] J. Hawkinson, "RFC1930 Guidelines for creation, selection, and registration of an Autonomous System (AS) Status," 1996, [Online]. Available: https://tools.ietf.org/html/rfc1930.

[60] Bill Dong, "Forbidden sites hijacked all over China," Tech. Rep., 2002. [Online]. Available: http://en.minghui.org/html/articles/2002/10/7/27311p.html.

[61] Hikinggfw.org, *THE GFW BLOCKED KEYWORDS*. [Online]. Available: https://hikinggfw.org/blocked%7B%5C_%7Dkeywords (visited on 03/10/2020).

[62] OpenNet, "OpenNet Initiative: Bulletin 005," Tech. Rep., 2004. [Online]. Available: https://opennet.net/bulletins/005/.

[63] Onoketa, *[TLS] Possible blocking of Encrypted SNI extension in China*, 2020. [Online]. Available: https://mailarchive.ietf.org/arch/msg/tls/YzT5LjLJ%7B%5C_%7D6WWhdnU2wVsKNKR6%7B%5C_%7DI/ (visited on 10/30/2020).

[64] Marc Bevand, *Experience With the Great Firewall of China*, 2016. [Online]. Available: http://blog.zorinaq.com/my-experience-with-the-great-firewall-of-china/ (visited on 11/01/2020).

[65] S. Chen, R. Wang, X. F. Wang, and K. Zhang, "Side-channel leaks in web applications: A reality today, a challenge tomorrow," *Proceedings - IEEE Symposium on Security and Privacy*, pp. 191–206, 2010, ISSN: 10816011. DOI: 10.1109/SP.2010.20.

[66] S. Hao, J. Hu, S. Liu, T. Song, J. Guo, and S. Liu, "Improved SVM method for internet traffic classification based on feature weight learning," *ICCAIS 2015 - 4th International Conference on Control, Automation and Information Sciences*, pp. 102–106, 2015. DOI: 10.1109/ICCAIS.2015.7338641.

[67] R. Yuan, Z. Li, X. Guan, and L. Xu, "An SVM-based machine learning method for accurate Internet traffic classification," *Information Systems Frontiers*, vol. 12, no. 2, pp. 149–156, 2010, ISSN: 13873326. DOI: 10.1007/s10796-008-9131-2.

[68] Z. Deng, Z. Liu, Z. Chen, and Y. Guo, "The random forest based detection of shadowsock's traffic," *Proceedings - 9th International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2017*, vol. 2, pp. 75–78, 2017. DOI: 10.1109/IHMSC.2017.132.

[69] Y. Zhao, Y. Yuan, Y. Wang, Y. Yao, and G. Xiong, "Evaluation scheme for traffic classification systems," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8710 LNCS, pp. 258–264, 2014, ISSN: 16113349. DOI: 10.1007/978-3-319-11119-3_24.

[70] D. Chepenko, *Demystifying the Great Firewall of China.* 2020. [Online]. Available: https://medium.com/mobile-asia/demystifying-the-great-firewall-of-china-22f4a97550cc (visited on 11/01/2020).

[71] V. Ni, "China Telecom, China Unicom Face Anti-Monopoly Probe," *China Briefing*, 2011. [Online]. Available: https://www.china-briefing.com/news/china-telecom-china-unicom-face-anti-monopoly-probe/.

[72] M. Li and Y. Zhu, "Research on the problems of interconnection settlement in China's Internet backbone network," *Procedia Computer Science*, vol. 131, pp. 153–157, 2018, ISSN: 18770509. DOI: 10.1016/j.procs.2018.04.198. [Online]. Available: https://doi.org/10.1016/j.procs.2018.04.198.

[73] R. Dingledine, "Tor: The Second-Generation Onion Router Roger," ISSN: 0038-531X. DOI: 10.1007/bf01118387.

[74] R. Jagerman, W. Sabée, L. Versluis, M. de Vos, and J. Pouwelse, "The fifteen year struggle of decentralizing privacy-enhancing technology," no. April, 2014. arXiv: 1404.4818. [Online]. Available: http://arxiv.org/abs/1404.4818.

[75] Torproject.org, *Torproject.org Blocked by GFW in China: Sooner or Later?* 2008. [Online]. Available: https://blog.torproject.org/torprojectorg-blocked-gfw-china-sooner-or-later (visited on 09/07/2020).

[76] ——, *China blocking Tor: Round Two*, 2010. [Online]. Available: https://blog.torproject.org/china-blocking-tor-round-two?page=1 (visited on 09/07/2020).

[77] ——, *Bridge easily detected by GFW*, 2011. [Online]. Available: https://trac.torproject.org/projects/tor/ticket/4185 (visited on 09/08/2020).

[78] ——, *Tor: Pluggable Transports*. [Online]. Available: https://2019.www.torproject.org/docs/pluggable-transports.html.en (visited on 10/21/2020).

[79] Y. Angel, *Obfs4 specification*. [Online]. Available: https://gitlab.com/yawning/obfs4/blob/master/doc/obfs4-spec.txt.

[80] N. Mathewson, *Ntor protocol*, 2011. [Online]. Available: https://gitweb.torproject.org/torspec.git/tree/proposals/216-ntor-handshake.txt (visited on 08/09/2020).

[81] Seth Elliott, *Iperf bandwidth testing software*. [Online]. Available: https://iperf.fr/ (visited on 11/30/2020).

[82] Torproject.org, *Tor Metrics*, 2020. [Online]. Available: https://metrics.torproject.org/ (visited on 11/01/2020).

[83] X. Xu, Z. M. Mao, and J. A. Halderman, "Internet censorship in China: Where does the filtering occur?" *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6579 LNCS, pp. 133–142, 2011, ISSN: 03029743. DOI: 10.1007/978-3-642-19260-9_14.

[84] Torproject.org, *Tor 0.2.3.9-alpha is out*. [Online]. Available: https://blog.torproject.org/tor-0239-alpha-out (visited on 12/06/2020).

[85] R. Ensafi, P. Winter, A. Mueen, and J. R. Crandall, "Analyzing the Great Firewall of China Over Space and Time," *Proceedings on Privacy Enhancing Technologies*, vol. 2015, no. 1, pp. 61–76, 2015. DOI: 10.1515/popets-2015-0005.

[86] Tor Project, *Test obfs4 reachability*. [Online]. Available: https://trac.torproject.org/projects/tor/ticket/29279 (visited on 11/17/2020).

[87] Alice, Bob, Carol, J. Beznazwy, and A. Houmansadr, "How China Detects and Blocks Shadowsocks," pp. 111–124, 2020. DOI: 10.1145/3419394.3423644.

[88] M. Krzywinski, "Port Knocking: Network Authentication Across Closed Ports," pp. 12–17, 2003.

[89] Linux Foundation, *Open vSwitch*. [Online]. Available: https://www.openvswitch.org/.

[90] Ryu-sdn.org, *Ryu SDN Framework*, 2017. [Online]. Available: https://ryu-sdn.org/ (visited on 12/24/2020).

[91] A. Narayanan, "A critique of port knocking," *Linux Journal*, 2004. [Online]. Available: https://www.linux.com/news/critique-port-knocking/.

[92] M. H. Eldefrawy, M. K. Khan, and K. Alghathbar, "One-time password system with infinite nested hash chains," *Communications in Computer and Information Science*, vol. 122 CCIS, pp. 161–170, 2010, ISSN: 18650929. DOI: 10.1007/978-3-642-17610-4_18.

[93] F. H. Mohd Ali, R. Yunos, and M. A. Mohamad Alias, "Simple port knocking method: Against TCP replay attack and port scanning," *Proceedings 2012 International Conference on Cyber Security, Cyber Warfare and Digital Forensic, CyberSec 2012*, pp. 247–252, 2012. DOI: 10.1109/CyberSec.2012.6246118.