Fall 2020 11/30/2020

Research report: Internet censorship and ways to circumvent it

Alexandre ROBIC CS 380: Intro to Cryptography

Censorship has a long history in almost every country, including the United States and has always found ways to adapt, even with the introduction of the most revolutionary method of communication: the internet. During WW2, the federal government set up the Office of Censorship with the objective of preventing confidential information from being leaked from the front lines to the Japanese and German. This was mainly achieved through the verification of letter contents sent from the front line as well as through self-enforcement for newspapers who were issued guidance about certain topics. All this was to prevent psychological operations from being effectively carried out either in the US by Japan or the UK by Germany but such operations are still underway today with Chinese media actively promoting civil unrest on the border with India, especially in the Aksai Chin and Arunachal Prundesh regions. Censorship is also a useful too to prevent revolutions and keep authoritarian regimes in place. A good example was during the Arab spring, in Libya when the internet was shut down by the government, forcing people to search their basements for old modems and telnet compatible computers in order to propagate the spirit of the revolution which will eventually bring Khadhafi to his end. Limiting what information is published on the web is also done by private companies like Facebook, Twitter or YouTube who decide what content gets to stay on their platform. Such behavior has however led to many criticizing big companies and blaming them of doing certain politicians' betting.

A prominent case against censorship can be the one against China. Since the 1990s in the wake of the Tiananmen square events, the world's second economy has taken drastic measures to prevent the spread of information and avoid another incident of such an international span. With the rise of the internet in the 1990s, the Chinese Communist Party (further abbreviated by CCP) has enforced new laws on which they regard as "internet security" while aggressively conducing cyber intrusions in foreign networks. This new method of communication, rather than bring democracy was to be used to control the population. During a presidential visit to Beijing in 1998, President Bill Clinton would go on to comment on the "manifestations of freedom in China" in front of an auditorium full of students who "sat there trying to figure out what the cost of applauding might be" in a country that the president described as being full of people able to speak their minds" [a].

In an effort to remain anonymous and avoid having to “register under their real identities for online forums and message boards” [b], many have turned towards technological novelties in order to freely communicate or access information without fear of their leaders. This report shall therefore explore different technologies available to counter censorship through papers written by various authors from world known conferences like ACM CCS (Association for Computing Machinery conference on Computer and Communications Security) or NDSS (Network and Distributed System Security Symposium). Such papers examine, analyze but also contribute to expand on topics like packet manipulation, routing, TLS, parasitizing and the decentralization of censorship networks.

Geneva: Evolving Censorship Evasion Strategies [1]

Many countries engage in censorship through in-network monitoring and censoring forbidden keywords (looking up “Tiananmen square massacre 1989” will get you banned from the internet in China for example). One way to circumvent on-path censorship is to perform client-side packet manipulation to confuse censors into not closing connections.

Ever since the democratization of the internet, a “cat and mouse game” has been going on between governments trying to prevent access to certain information and citizens seeking the real narrative. Discovering new censorship evasion methods is long and requires a lot of manual work to better understand the network you’re trying to circumvent the censorship on and by the time a censorship evasion strategy in place, the censor network can have already evolved.

Geneva is a genetic algorithm that "automatically evolves censorship strategies against nation-state censors". It takes advantage of fact censors employ incomplete implementations of network stacks. Geneva runs on client side and does not require anything extra from hosts. It also does not disrupt the traffic on a censored network for other users.

The genetic algorithm part comes from fact that Geneva discovers packet manipulation strategies on its own. The technical challenge of genetic algorithm design is balancing strategies it can find: either let it perform arbitrary bit manipulations which allows it to learn all strategies but also becomes computationally expensive or only perform packet manipulations discovered in previous work which may lead to the rediscoveries of obsolete techniques.

Geneva was tested again the GFW (Great Firewall of china), India and Kazakhstan. Across the 27 experiments conducted on the GFW, Geneva discovered 4 unique species, 8 subspecies and 21 variants of subspecies of algorithms that can be used to circumvent that specific network.

To reach its objectives, Geneva must go through two main types of censorship strategies:

On path censorship can be seen as a big state sponsored MITM on a network in order to process every packet. This represents a stronger attack model and has the "power to manipulate or drop traffic". It is extremely computationally expensive to run and can create network bottlenecking so most of GFW is done through "man on the side" components.

Man-on-the-side systems can still view packets but sit outside network and cannot modify or drop traffic. To prevent access to unauthorized information, the system sends a TCP RST packet (packet sent by TCP sender to indicate it will no longer accept nor send more data).

By default, if a packet is not going to a blacklisted IP or the deep packet analysis does not reveal anything suspicious, it'll be allowed to continue. Packet-manipulation based evasion is a "sequence of packet manipulations (modifying, adding or dropping) to evade a censor".

Geneva's design revolves around genetic algorithmic which can be described as a biologically inspired approach to automate algorithm design. It requires 3 components: a series of genetic building blocks to represents different algos, a fitness function to capture how well a given algo performs and methods for mutating and crossing over to generate new algorithms. It is based around 4 operations (or functions):

-Drop: simply drops the incoming packet

-duplicate(a,b): copies a packet and applies action sequence a to the original packet and b to the copy

-fragment(a, b): segments a packet at a specific offset byte and applies action sequence a to the first part and b to the second part

-tamper(a): alters the given field of a packet and applies action sequence a (either replaces the given field to a new value of the same bitlength or corrupts it by setting a random value of the same bitlength)

Through an iterative process of generation and selection, genetic algorithms simulate Darwinist selection (hence the genetic aspect) by taking the fittest algorithms and crossing them with each other to obtain the best one:

In order to evaluate a strategy, Geneva will make a censorship-worthy (like with a banned keyword or to a blacklisted IP) GET request using the strategy and its fitness will be determined by whether a request can be completed. If it can, the fitness is positively increased, otherwise, it is very largely decreased.

After each step (doing all the requests and evaluating each strategy), Geneva runs a "selection tournament": it takes the methods with the highest fitness and adds them to a set called the "offspring pool". This repeats until the offspring pool is the same size as the population pool (or when each individual method has been replaced with one of the best ones).

Geneva was implemented in about 6000 lines of Python and runs on the client and uses NetfilterQueue to access the client's packets.

It was experimented on a series of 11 mock censors (fake censorship networks created to imitate real life implementations) and included the injection of TCP RST packets (to simulate China) and in-path censors dropping packets (to simulate India and China) among other techniques.

Although it proved effective against the 11 virtual mock censors, it still needed to be tested against real life censorship networks and some questions still needed to be solved like what strategies it would find against censors and if it can be applied to several censorship networks.

Geneva discovered successful strategies in 23/27 experiments against the GFW (it fails to discover more strategies when given limits like restricted access to TCP headers). One strategy it has discovered for example, is to simply drop the incoming RST packets after the censor discovers a forbidden word in the sent request or in the incoming result.

How can censors avoid Geneva? According to paper [1], by fixing bugs that allow for the censorship evasion strategies to be found to detect the training packets Geneva sends out which would poison its datasets.

The increased difficulty in obtaining VPSs in mainland China by non-Chinese residents (to avoid this kind of experimentation) made it difficult to further the experiment against the GFW. The initial servers could only be obtained through the help of “anonymous help”. The authors of the project paper believe Geneva is the “first step towards automating censorship evasion” and to keep the initiative going, have made their code and data publicly available: https://geneva.cs.umd.edu.

Although the use of genetic algorithmics seems like a good idea to circumvent censorship, China has seen its government increasingly sponsor technological projects like facial recognition or the famous social credit system. Just like other intelligence agencies, the MSS (Ministry of State Security, China’s foreign intelligence services) heavily monitors new innovations in fields that might be relevant to its objectives like that of censoring material that may damage the regime. After reading such a paper and having access to the code, it wouldn’t take long for hackers based in Pudong and especially PLA unit 61398 to reverse engineer it and use it to protect and ameliorate the very network it needs to take down.

From a user standpoint, it seems like Geneva is still in its relatively early stages of development. No software based around the methods found exists, making it impossible to be commercialized or installed on a consumer machine.

Decoy Routing Circumvention that Resists Routing attacks [3]

As seen in the previous paper, many regimes censor their citizens. Sometimes the solution to preventing the censorship is to simply avoid the censorship network using VPNs or Tor. The only issue with that is RAD attacks (Routing Around Decoys) which forcefully route traffic through the censorship networks to avoid the use of decoy networks. The paper proposes a new routing architecture that can circumvent RAD attacks. To achieve this, the authors introduce a new system of decoy routing that involves only operating on the downstream (response from the server). This contrasts the previous routing architectures that that intercept the upstream traffic of the censored users. Deploying only "downstream-only decoy routing is a major step forward in making decoy routing systems practical" since their number can be reduced as opposed to upstream decoy routing.

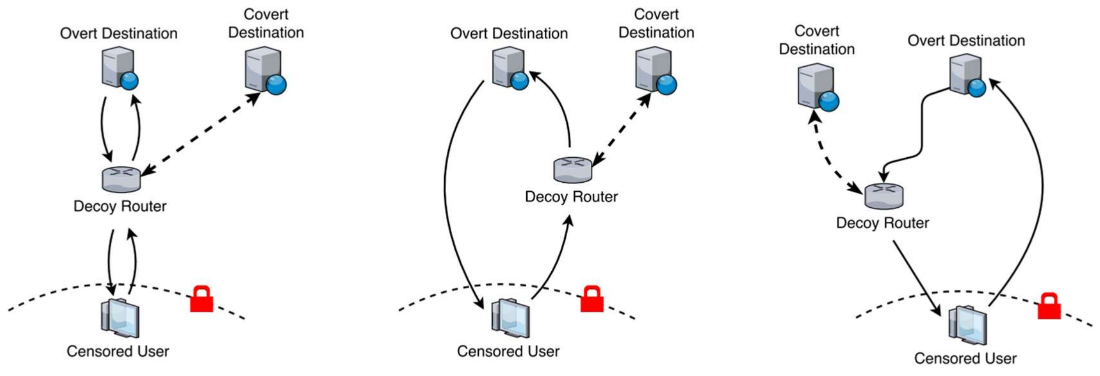
Designing a downstream only decoy routing system poses a few engineering problems, mainly the fact that previous decoy routing systems use the upstream traffic of censored users to communicate covert messages like "HTTP GET requests for blocked destinations".

The authors introduce Waterfall, "the first downstream only decoy routing system" that makes use of HTTP redirection mainly but also other techniques.

Traditional routing methods like Tor or VPNs rely on the use of proxies on networks outside of that of the censored user and are used to bypass censorship by forwarding the traffic through the various servers. Tor for example, has "several thousands of volunteer proxies", servers set up by individuals to be used in the routing network. The downside to the use of such proxies is that the censorship network can start blocking IPs included in the network which will eventually lead to it no longer being available in the censored country through various attacks (probing, insider sabotage etc). Another solution implemented is Domain fronting which involves the implementation of web services on the same IPs with "other non-circumvention services", thus preventing the censors from blocking that node based on its IP.

Decoy routing is a recent approach to censorship circumvention with the intention of preventing IP blocking by censoring countries. To do such a thing, it relies on ASes (or internet Autonomous Systems) that modifies its routers to deflect the traffic of censored users to the blocked destinations. In theory, "decoy routing defeats IP address blocking".

The architecture offered looks like the following (see figure c):



(a) Upstream (Type I) [21, 25, 59] (b) Upstream (Type II) [4, 15, 58] (c) Downstream-Only

In this configuration, the decoy router does not need to intercept the upstream overt traffic. This makes it appear as completely legitimate traffic to the eyes of the censor but, hides covert traffic.

Routing Around Decoy attacks (or RAD) rely on being able to find and "discard upstream routes" that target known ASes and reroute traffic through routes impossible to decoy. The objective of such attack is to "prevent censored users from using decoy routing systems" by routing their traffic around said decoys. It's a costly attack for censoring regimes as it needs to provide a decoy-free route to the destination or make it unreachable (and risk making non censored destinations unavailable). In theory, downstream only routing is meant to prevent such attacks on its use.

The idea of downstream only routing decoy routing is to offer a stronger resistance to RAD attacks by operating decoy routers only on the downstream traffic of censored users. Since censoring regimes have more control over upstream traffic and can reroute that traffic more easily, Waterfall thus offers a stronger approach to censorship evasion.

Waterfall's design resides around several steps:

-Establishing an overt connection: An overt TLS connection with a non-blocked destination is established. Unlike previous designs, this traffic does not need to be routed through a decoy AS.

-Authentication: Waterfall decoy routers need to identify the registered clients' traffic. This allows for protection of clients' confidentiality as well as prevent sabotage by non-registered users.

-Covert communication: a client can then communicate with the decoy router through downstream covert channels.

Several experiments were conducted to measure how efficient the downstream routing is and seem to show a steady relationship between the number of kB received and the overt pages download time which would point towards a very resource friendly system and a resistance to traffic analysis attacks. To resist such attacks, the decoy routing system needs to preserve certain patterns. If censor discovers there's a certain delay between the time taking to request a page from the region the user is in and the time taken to receive the information by the user (the user's information takes slightly longer), the censor may get suspicious.

Overall, Waterfall seems like a novel architecture to be put in place as it can more easily circumvent censorship networks. The issue is, as described in the previous paper and in the introduction, censorship evasion is a constant cat and mouse game and it is likely censorship networks will catch on to downstream decoy routing. As described too in the research report, it's mentioned several times such an architecture is complicated to be put in place, making it less likely to see the day as of today.

The use of TLS in censorship circumvention [2]

TLS (or transport layer security) is "the most popular protocol on the internet" for security and interaction between clients and servers and is used by a good 70% of pages on Firefox. Because of its widespread use, it's distinguished itself by using several implementations and being constantly changing for security reasons. This wide range of features makes it possible to distinguish implementations from one another (cipher suites, elliptic curves, signature algorithms etc). Deep packet inspection has been able to identify and block popular TLS tools based solely on their fingerprints.

In response to this blocking, many tools attempt to mimic popular TLS implementations by mimicking their fingerprints in order to get passed censors. This technique, although effective to a certain degree, poses a wide variety of problems, mainly by the fact TLS is constantly changing for security reasons and implementations are difficult to mimic perfectly.

The paper introduces the collection and analysis of "real world TLS traffic from 11.8 billion connections over 9 months" from students at the university of Boulder in Colorado to identify the TLS implementations actually used on the internet but most importantly identify the TLS implementations of popular censorship circumvention tools like Signal or Tor.

Censors can easily block custom protocols since TLS handshakes are not encrypted. Censors can therefore identify key exchange algos, extensions in a Client Hello message. Popular tools like Tor were blocked since they used specific SSL/TLS features.

Signal employed domain fronting to evade censorship in several Arabic countries that practice censorship, but its fingerprints were easily distinguishable from the TLS implementations they were trying to mimic.

TLS is therefore difficult in making implementations robust against censorship.

The study was conducted on TLS connections over a 10Gbps tap at University of boulder CO (11B connections over 9-month period). To better group implementations, researchers hashed the unchanging parts of the Client Hello message to group connections made with same implementation and stored everything in a database.

As well as looking at connections from students and faculty to everyday websites, they also looked at apps like Signal, Lantern or Snowflake (anonymous encrypted communication). The data is available online at https://tlsfingerprint.io.

Authors created the uTLS library to mimic popular TLS implementations and integrated data from the dataset from the University of Boulder to better automatically generate code.

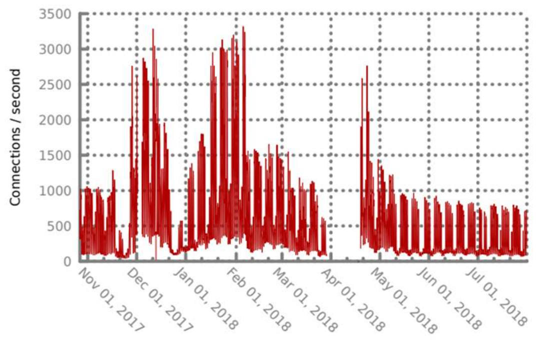
TLS handshakes are unencrypted and allows to authenticate identities, agree on keys etc which makes it easier to analyze. A quick experiment with a packet sniffing tool like Wireshark on the school's network can reveal what is in a handshake, mainly client IP, the server IP and the port being used. Analysis of such data can reveal what a user is doing and where (port 443 can reveal a web search and 22 can show a user was using SSH to connect to a machine).

3 kinds of information were collected during the experiment:

Client Hello messages: extracted TLS version, handshake version, cipher suites and extensions. All this hashed and fit to a 64-bit format was then stored in a database and the number of times that specific fingerprint appeared in connections was recorded.

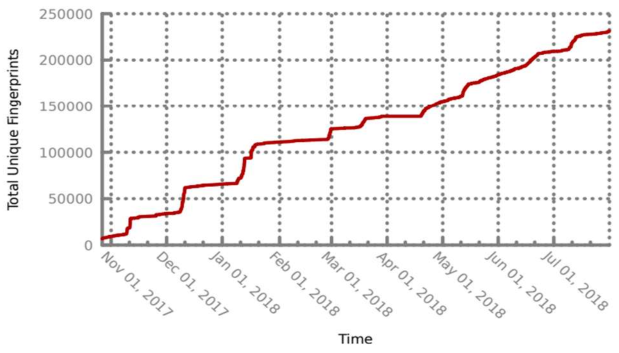
Connection specific information: includes an anonymized IP, server name which allows to keep track of sources

Server Hello messages: allowed to better parse TLS record version, see what cipher suite and extensions were negotiated successfully, compression etc.



Number of connections observed during the 9-month period. Dips can be seen during weekends and breaks (mainly spring break in April).

Results:



The number of unique fingerprints increased during the year with certain stagnations (explained in the paper by a small set of internet scanners sending random ClientHello messages).

Another thing that can explain the sudden increases in fingerprints is the updates and new versions of web browsers that are released often.

What this experiment wants to show is that by collecting enough information about TLS implementations, one can easily discover the origin and the implementation thus linking a certain fingerprint to an application or a source to be more easily censored.

This was verified when the authors analyzed both the iOS and Android version of Signal and found that on iOS it generates a native iOS fingerprint which they concluded "unlikely a censor could block Signal on iOS from the ClientHello fingerprints alone". On Android however, they found Signal only generates 4 distinct fingerprints that were not seen in the rest of the dataset, making it easy to detect.

Lessons learned:

To better hide from censors, you can try mimicking a TLS implementation but need to find a popular one (choose popular web browser according to paper).

The problem is implementations can change overtime so there’s a need to keep up or the difference can become visible.

One can randomize TLS Client Hello messages which means there’s no need to identify or track support for TLS implementations, but this can only work if it mimics distribution of TLS implementations (stays within statistics).

Parasitizing as a mean to evade censorship: Protozoa’s approach [5]

Many censorship circumvention tools rely on proxies that allow users to go through covert channels and access it through tunnel like the use of skype calls. There’s a need to find a compromise between good bandwidth to allow the use of regular internet activities by regular users but also be secure against traffic analysis attacks. The authors present protozoa, a censorship resistant tunneling tools that features both high performing covert channels as well as traffic analysis resistance. This is done using a videocall with a service available outside and inside the censored region.

With the covid outbreak, many news sites have been censored in China and keywords on social media platforms related to the infection. This is done through keyword filters, image filters, social media monitoring and even IP blocking certain destinations and the protocols used to reach them.

Many tools are offered to avoid censorship, a lot rely on covert channels through the transmission of data by an overt multimedia platform. The goal is to be able to encode the data in such a way the censorship network can't distinguish between a legitimate communication and a covert one. This is called multimedia covert streaming and is achieved through 2 ways:

-Mimicking the entire application's network protocols (media protocol mimicking)

-Embedding covert data into video or audio signal (raw media tunneling)

The censorship network can choose to ban such services all together, but some are essential in maintaining economic and social ties between countries, which would result in a terrible political decision. Another thing an adversary can do is probe the traffic generated by the video calling application and look for abnormalities in traffic which might signal the presence of covert channels. If picking from 2 channels (one covert on and one overt one), an adversary should not be able to distinguish both channels in more than 50% of the time.

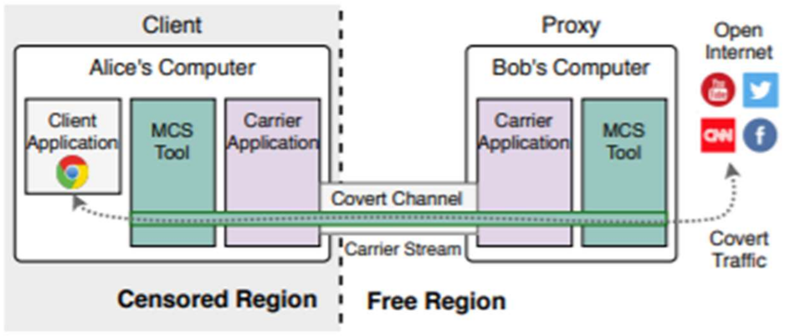
An experiment conducted shows that protozoa was able to be discovered in 55-65% of cases compared to other similar tools which were discovered in network analysis attacks in over 99% of times.

A media covert system works by enabling the client to overcome the restrictions enforced in the censored region by making use of a proxy located in a free region operated by a trusted user and by using a carrier application consisting of an encrypted video stream.

In order to defeat a MCS systems, an adversary can use deep packet inspection to locate indicators that may point towards a covert channel being used but can also launch active network attacks in order to disrupt the activity of regular channels and better identify the covert ones since they won't react the same way.

Protozoa's architecture:

Follows the general MCS tools architecture:



Generally, an application will consist of a backend that handles covert communication, session establishment and a client-side HTML page that initiates video call and manages video transmission via the web browser.

The MCS tool materialized by Protozoa is a software bundle that targets the Linux platform and be setup as a client or a proxy depending on the party:

-The client starts by using a regular IP application

-The traffic is encoded and routed through the WebRTC covert channel

-When it arrives at the proxy, it's sent to a decoder and then forwarded to the censored destination

-When the server response is returned, it'll then be encoded (to avoid being detected and for the connection to be interrupted)

--The retrieved packet is then sent to the censored user through the covert channel

-The software will then decode it and send the packet to the IP application for viewing

Protozoa employs what the authors call "Encoded media tunneling" as opposed to raw media tunneling. Rather than replace the pixels of the raw video input, it replaces bits of the encoded video signal after compression. According to the authors, this helps increase the capacity of the channel but also makes it resistant to traffic analysis attacks.

Protozoa evaluations of performance were based on 2 main criteria:

Security: being able to transmit data without being detected by the censoring network (a local censorship network was setup for the experiment).

Performance: being able to transmit and receive data efficiently (measurement of ratio between total amount of data transmitted and total amount of available space in encoded video frames).

Results:

For security, the true performance rate generally followed the false performance rate which points at a secure application since the difference between the times it was correctly identified and the times it wasn't is minimal.

For performance, it was able to keep up between 1250 and 1500 kbps of throughput in the lab conditions.

Overall, it requires a constant supply of volunteering proxies in order to prevent collapse of network (if too many connections from individuals go through a certain proxy, it might end up being blocked). Protozoa also demonstrates an increase of in throughput of up to 3 times the regular throughput of a regular raw media encoding application.

Decentralization of censorship networks: The Russian approach to censorship [4]

Network control has been a goal of nation states since the internet became popular worldwide. It's been noticed that the more use of internet and social media, the more political tensions rose (ex: Russian election of 2018 when social media relayed ballot stuffing happening all over the country).

Countries struggle to effectively control internet and sometimes resort to simply shutting down the internet in the entire country or certain regions, an unpopular decision which has become the de facto method of censorship for several countries when political tensions got to high. This generally prevented activists from coordinating.

2 main methods of censorship have emerged throughout the past 20 years:

-Centralized censorship like the great firewall of China or in Iran and involves an internet chokepoint.

-Decentralized networks: the example studied here with Russia, a difficult method to setup since ISPs must be coordinated which they are rarely.

Russian policies have allowed more and more control of media these past years due to Putin's appearance with self-representation. In classic dictatorial fashion, cannot support criticism and opposition (recent poisoning of political opponent Alexei Navalny).

The team behind the paper spent a year talking with activists inside Russia to obtain 5 lists of IPs, domains and subnets blocked in Russia by ISPs signed by the Roskomnadzor (the Federal Service for Supervision of Communications, Information Technology and Mass Media). Help from inside the country also helped gain access to a VPS within Russia in order to better understand what is blocked and how it is blocked depending on each ISP.

Measurements were performed by activists from 20 different servers and with additional vantage points by using tools like Quack and Satellite that can easily detect interferences in application layer networks.

Experiments show not all ISPs block content in similar ways, but number of websites blocked in residential areas is very high. This blocking is done mainly though TCP layer blocking, application layer blocking (hence the use of Quack for the experiment) and deep packet inspection.

Background work:

This was mainly done on centralized and decentralized censorship networks:

Centralized: China and Iran use that kind of system since the government owns ISPs or at least has a huge control over it. This allows control of information on very high scales but can "have dramatic effects" if there are even "small perturbations". The example cited is that of North Korea's only ISP link with China’s Unicom being closed which resulted in a loss of internet throughout the entire country. Censors like to apply even mix of censorship methods on network to avoid users focusing on avoiding one and does this on the entire network stack.

Decentralized: Recently, several countries have been using this system since they don't have the same kind of control over their ISPs like China or Iran. Examples include the UK and Russia who have passed laws asking ISPs to block certain content (anti-government content in the case of Russia and Jihad material in the UK). The methods used to achieve these goals vary from country to country, with places like Indonesia relying heavily on DNS manipulation and India using a combination of DNS manipulation, HTTP filtering and TCP/IP blocking.

Censoredplanet.org, the main sponsor of this study has been focusing on 3 types of censorship techniques:

TCP/IP blocking: This consists in the censor disrupting communication by blacklisting the IP address of a user or another entity needing to be censored for political reasons. It's the cheapest, most common and most effective way of censoring unwanted information and can cause huge collateral damage for websites hosted on the same IP address. This method is mainly used in Iran and China to block Tor relays and other proxies.

DNS manipulation: A censorship network can observe DNS queries to certain domains and decide to return a "host not found" error message or point towards a web page explaining why a certain page was blocked (generally called a block page). This can easily be circumvented by using an alternate DNS resolver like Google's famous 8.8.8.8 (which has since been banned in places like China).

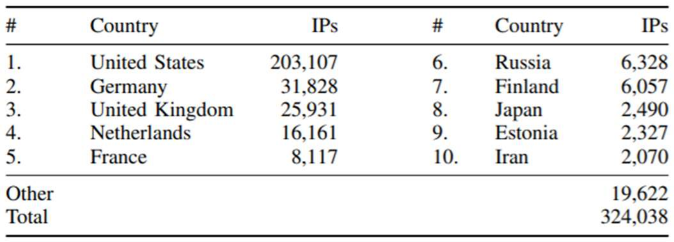
Keyword based blocking: This involves using deep packet inspection to scan a request for keywords that may lead to banned content or historical narratives a country does not want made publicly available. If such keywords are found, the server can send a TCP RST package or point towards a block page.

Experiment design:

The preparation of experiment resulted in discovery of leaked block lists that the Russian ISPs would receive regularly from the Roskomnadzor. The one used for the experiment dates to April 2014 and was nicknamed RUBL (RUssian BLocklist). It was discovered that RUBL contained a total of "324,695 unique IPs, 132,798 unique domains, and 39 mutually exclusive subnets". Using the various vantage points scattered in Russia, censoredplanet.org was able to discover the nature of the IPs and domains blocked as well as gain a better understanding of how to requests were managed and resolved.

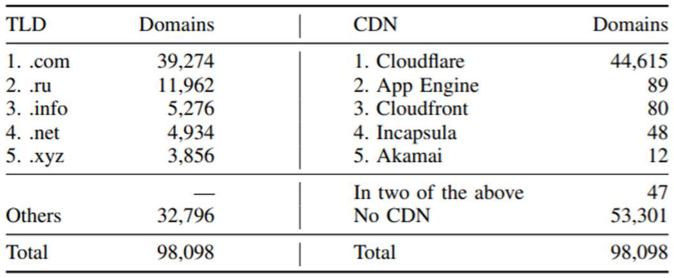
Results:

These tables show the origin of the websites and IPs in the RUBL:



#	Country	IPs	#	Country	IPs
1.	United States	203,107	6.	Russia	6,328
2.	Germany	31,828	7.	Finland	6,057
3.	United Kingdom	25,931	8.	Japan	2,490
4.	Netherlands	16,161	9.	Estonia	2,327
5.	France	8,117	10.	Iran	2,070
Other					19,622
Total					324,038

Table II: **Top ten countries hosting IPs on the blocklist.** ◇



TLD	Domains	CDN	Domains
1. .com	39,274	1. Cloudflare	44,615
2. .ru	11,962	2. App Engine	89
3. .info	5,276	3. Cloudfront	80
4. .net	4,934	4. Incapsula	48
5. .xyz	3,856	5. Akamai	12
	—	In two of the above	47
Others	32,796	No CDN	53,301
Total	98,098	Total	98,098

Table III: **Top five TLDs and CDNs for domains in the blocklist**—.com and .ru are the most popular TLDs. ◇

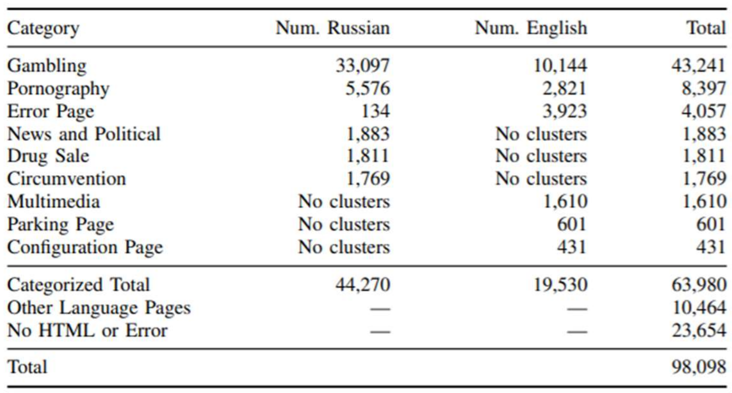
Some of these results are not at all surprising:

The USA has long been both a political and economic rival of Russia with its leaders wanting to see a government system more closely based on a western democracy.

Russian pages and IPs being censored can be from political opponents and other organizations that the regime does not want to see spread information.

Estonia, ever since the annexation of Crimea in 2014, has increased its arsenal and tightened its links to NATO. This anti-Russian behavior can explain the blocking of many IPs on the Estonian network.

Within the blocked webpages, around 72% were exploitable in the study. Within those 72% of documents, it was found that a good 63% were in fact in Russian or in Cyrillic alphabet, thus being intended to a Russian speaking audience and 28% in English to targeting a more global audience. The 2 most popular domain signatures were .com (40000 domains) and .ru (12000), pointing at a significant censorship of content from the Unites States but also a very large number of internal websites.



Category	Num. Russian	Num. English	Total
Gambling	33,097	10,144	43,241
Pornography	5,576	2,821	8,397
Error Page	134	3,923	4,057
News and Political	1,883	No clusters	1,883
Drug Sale	1,811	No clusters	1,811
Circumvention	1,769	No clusters	1,769
Multimedia	No clusters	1,610	1,610
Parking Page	No clusters	601	601
Configuration Page	No clusters	431	431
Categorized Total	44,270	19,530	63,980
Other Language Pages	—	—	10,464
No HTML or Error	—	—	23,654
Total			98,098

This table shows the category of websites blocked by domain, with gambling being at the top. The reason behind that can be the oligarchy Russia faced between the fall of the Soviet Union and Putin's arrival in power with gambling being used as a common front for money laundering.

Overall, Russia is an excellent case study for censorship since it uses a decentralized network as opposed to China or Iran which are more commonly studied. The effectiveness of such decentralized censorship networks can raise important questions especially regarding countries that would be looking at implementing their own internet censorship (perhaps to a lesser level than Russia). Seeing as the censorship is done through various networks and not even one, censorship evasion strategies are much more difficult to find since they need to be effective against every network and not a single firewall. Russia's censorship architecture could be reproduced throughout the world in the next years by countries looking to limit access to certain content, some for political reasons, other for legal reasons (access to "adult content" in certain regions for example).

Pros and Cons of each censorship evasion method, tradeoffs and comparison

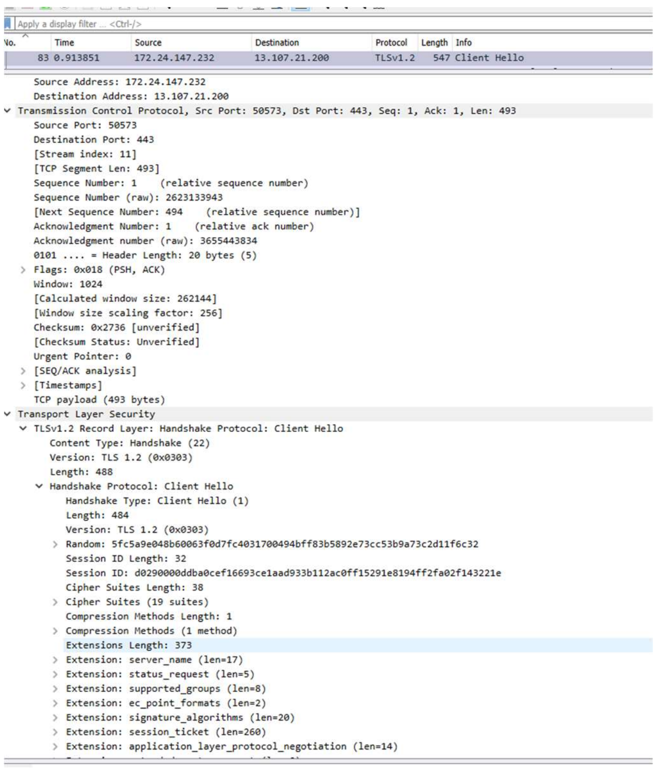
Geneva: The problem with Geneva is it uses the same websites as targets (the top 1000 sites on the Alexa list) and this can easily be observed by the censorship network, that can simply decide to block all packets from a certain source to the targets in question. One main advantage of Geneva though, compared to the other solutions presented is that it's focused not on a single solution, but on a multitude of solutions using one same technique.

Protozoa: Protozoa seems like a novel idea since it relies on already available application for video calling rather than must create its own application that can easily be detected. The only downside to that is if a government decides to ban that specific application in favor of an application it has more control over (ex: China banning WhatsApp in favor of WeChat), Protozoa's implementation collapses.

TLS: TLS is a more complicated issue since it relies on several implementations which vary depending on the device, the web browser but most importantly, the version. One major downside is the lack of encryption of Client Hello messages which can easily be used to detect traffic going to unauthorized destinations as those packets contain the origin IP and destination of a packet as well as the port which can indicate the type of activity. Mimicking TLS fingerprints is also a complicated matter due to the ever-changing implementations and if the implementation doesn't fit into statistical uses (more TLS fingerprints for a certain app on a network than users actually connected to that app's servers), it can easily be detected. The advantage however of the multitude of implementations is the fact there's no way of blocking a single fingerprint since it updates constantly.

Waterfall: Although Waterfall presents itself as a downstream only proxy network architecture, it still requires an upstream covert channel to work correctly and initiate contact with the server which can easily be intercepted by the censorship network. One advantage it does offer, is it relies on a niche technology (downstream based proxy networks) to operate which has not been researched a lot by censorship networks.

One question we can genuinely ask ourselves is which one is the best? Looking at each solution, it's clear Geneva is only a proof of concept and not an implementable solution, it seems more like a first step in research and the last paper presented a case study which is not an implementable solution. This leaves us with 3 possible candidates: Protozoa, Waterfall and TLS. As TLS leaves easy fingerprints, it can automatically be disqualified (an experiment on the school network can easily show the intention of a user by analyzing a ClientHello message).



```
Apply a display filter ... <Ctrl-/>
No.   Time       Source           Destination     Protocol  Length Info
   83 0.913851   172.24.147.232   13.107.21.200   TLSv1.2   547 Client Hello

      Source Address: 172.24.147.232
      Destination Address: 13.107.21.200
v Transmission Control Protocol, Src Port: 50573, Dst Port: 443, Seq: 1, Ack: 1, Len: 493
      Source Port: 50573
      Destination Port: 443
      [Stream index: 11]
      [TCP Segment Len: 493]
      Sequence Number: 1    (relative sequence number)
      Sequence Number (raw): 2623133943
      [Next Sequence Number: 494    (relative sequence number)]
      Acknowledgment Number: 1    (relative ack number)
      Acknowledgment number (raw): 3655443834
      0101 .... = Header Length: 20 bytes (5)
    > Flags: 0x018 (PSH, ACK)
      Window: 1024
      [Calculated window size: 262144]
      [Window size scaling factor: 256]
      Checksum: 0x2736 [unverified]
      [Checksum Status: Unverified]
      Urgent Pointer: 0
    > [SEQ/ACK analysis]
    > [Timestamps]
      TCP payload (493 bytes)
v Transport Layer Security
    v TLSv1.2 Record Layer: Handshake Protocol: Client Hello
        Content Type: Handshake (22)
        Version: TLS 1.2 (0x0303)
        Length: 488
      v Handshake Protocol: Client Hello
          Handshake Type: Client Hello (1)
          Length: 484
          Version: TLS 1.2 (0x0303)
        > Random: 5fc5a9e048b60063f0d7fc4031700494bff83b5892e73cc53b9a73c2d11f6c32
          Session ID Length: 32
          Session ID: d0290000ddba0cef16693ce1aad933b112ac0ff15291e8194ff2fa02f143221e
          Cipher Suites Length: 38
        > Cipher Suites (19 suites)
          Compression Methods Length: 1
        > Compression Methods (1 method)
          Extensions Length: 373
        > Extension: server_name (len=17)
        > Extension: status_request (len=5)
        > Extension: supported_groups (len=8)
        > Extension: ec_point_formats (len=2)
        > Extension: signature_algorithms (len=20)
        > Extension: session_ticket (len=260)
        > Extension: application_layer_protocol_negotiation (len=14)
```

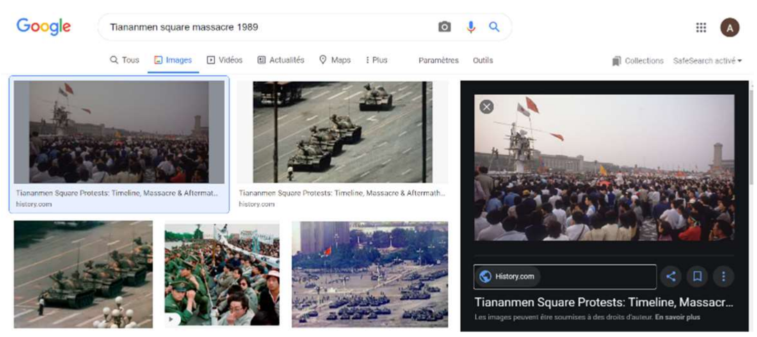
A captured ClientHello packet from the school's network (172.24.147.232 is the IP of the computer I'm writing this on and 13.107.21.200 is Microsoft's server, likely the backup of the research report).

This now leaves us with only Protozoa and Waterfall. Waterfall as described in the paper that introduces it is "difficult to put in place" and has not been implemented in an actual piece of software. A quick visit to the github page reveals that it's only in experimental stages and for the experiment to work, both the client and decoy need to be on the same network card (the same physical machine).

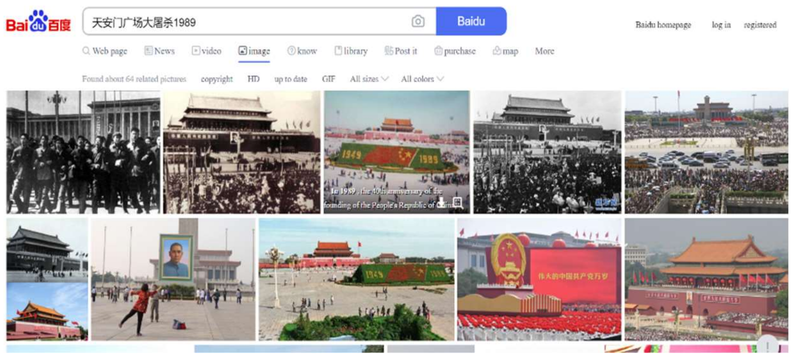
This eventually narrows it all down to Protozoa. As much at it seems like an effective solution, the only problem is that (for now, this will be corrected in the future) need to find the destination proxy on your own. Doing such may require overt communication with an exterior source which can completely defeat the purpose of the software if the censorship network finds out where the proxy is and IP bans it. Yet, it remains the best solution presented by the fact it can easily be used with a service available both in and outside the censorship region.

Conclusion

Sometimes, censorship isn't just about preventing people from finding out all information about a certain event but can also include a rewriting of history, a different narrative a government may try to push rather than completely delete a date and time. A good example of this can be the infamous Tiananmen massacre of 1989 which was the direct result of a protest turned nationwide that disrupted the visit by the Prime Minister of the Soviet Union, Mikhail Gorbachev and the 40th anniversary of the creation of the People's republic. On the 4th of June 1989, seeing that the presence of the military did not calm down tensions, live rounds were fired in the crowd. This event marked China's perception throughout the entire world as the protests were being broadcast live but most importantly lead to 2 narratives being pushed:


Google
Tiananmen square massacre 1989
Tous Images Vidéos Actualités Maps Plus Paramètres Outils
Collections SafeSearch activé
Tiananmen Square Protests: Timeline, Massacre & Aftermat...
history.com
History.com
Tiananmen Square Protests: Timeline, Massacr...

The historical one claiming the army fired live rounds into the crowds with the infamous photo of the man with his groceries in front of tanks on a street ironically named "avenue of eternal peace". He was eventually taken away by fellow protesters who feared he would get run over by the tanks.


Baidu百度
天安门广场大屠杀1989
Baidu
Baidu homepage log in registered
Web page News video image know library Post it purchase map More
Found about 64 related pictures copyright HD up to date GIF All sizes All colors

The official Chinese Communist Party narrative that the students were "evacuated" around 5AM and that by 5:30 the "entire clearance process was over" according to the Chronicle of the Chinese Communist Party.

A quick image search on Baidu.com, China's new alternative to Google ever since the company refused to host their google.cn servers on mainland China, illustrates this narrative rewriting very well. (A slowing down of the internet was also observed when the Baidu page was open).

Ever since those events, the CCP realized it had to better control the population and the introduction of the internet to the world certainly didn't help the cause of democracy in China.

Tiananmen square is not the only event that has shaped Chinese media censorship.

In 1997 when the 99 year rent of Hong Kong was up, Tony Blaire, then prime minister of the UK, signed over rights to politically control Hong Kong over to the People's Republic of China under terms that had been negotiated by Margaret Thatcher that included the right to free speech, free trade and the freedom to assemble. Since Xi Jinping's arrival in power in 2012 and since the construction of the world's biggest floating bridge from Macao to Hong Kong, protest movements have gotten more and more important with 2019 marking the height of the movement with up to 2 Million protesters for a city populated by 7 Million. This led the Chinese government to enforce the same kind of internet censorship the mainland faces, mainly through IP bans, censorship of keywords and censorship of non-Chinese search engines as well as cutting the internet or throttling it in the region in order to impose a siege on the population of Hong Kong and prevent it from assembling.

Another initiative to circumvent internet censorship is the use of videogames, especially the in-chat functions. As some games are not censored in certain areas of the world, it makes it easy for intelligence officers to communicate covertly in a game of CS: GO or the famous fortnite.

Overall, internet censorship remains a game of cat and mouse with censorship networks constantly upgrading their means in order to detect and prevent non-censorable traffic from going into and out of their region or country.

The internet was initially released to the world with the objective to make global communication easier but the global tendency of the 21st century seems to indicate the contrary. As more and more services move online (paying taxes, buying a bus pass among other necessities), it seems clear that abusive governments around the world are now seeing this tool as a way to easily spread their propaganda. As many companies are fond of the Chinese market (1.3 Billion people available), we might start to see American companies censoring content from domestic clients in order to have access to the foreign markets on the demand of the governments.

Papers used:

[1]. Kevin Bock, George Hughey, Xiao Qiang, Dave Levin. Geneva: Evolving Censorship Evasion Strategies. ACM CCS 2019. Link

[2]. Sergey Frolov, Eric Wustrow. The use of TLS in Censorship Prevention. NDSS 2019. Link

[3]. Milad Nasr, Hadi Zolfaghari, Amir Houmansadr. The Waterfall of Liberty: Decoy Routing Circumvention that Resists Routing Attacks. ACM CCS 2017. Link

[4]. Multiple Authors. Decentralized Control: A Case Study of Russia. NDSS 2020. Link

[5]. Diogo Barradas, Nuno Santos, Luís Rodrigues, Vítor Nunes. Poking a Hole in the Wall: Efficient Censorship-Resistant Internet Communications by Parasitizing on WebRTC. ACM CCS 2020. Link

Additional references:

[a] The perfect weapon by David E. Sanger

[b] CSIS website: https://www.csis.org/analysis/shrinking-anonymity-chinese-cyberspace