EVALUATION OF HTML TAG SUSCEPTIBILITY TO STATISTICAL FINGERPRINTING FOR USE IN CENSORSHIP EVASION

by

KELLY SCOTT FRENCH

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2016

To my family

I would like to express my deep gratitude to my parents, especially my mother, Sally Flora-French, who always encouraged me to dream big and to be willing to work for whatever it was I put my mind to. To my professor from Northeastern State University, Dr. Michael D. Bolton I wish to convey my sincere appreciation for his interest in my success and his willingness to share his advice to this willing student. And last but by no means least I would like to thank my wife, Donna, who supported me through the late nights, the scheduling headaches, and the many trips to campus, without her not only would I not have been able to pursue my academic dreams, more importantly, I would not be the man or father that I am today.

ACKNOWLEDGEMENTS

I would like to thank my thesis adviser Dr. Matthew Wright for all his help during my time at the University of Texas at Arlington. I appreciate the opportunity to work with him and learn as much as I could as I found lots of inspiration from how he conducts his classes to how he approaches research in general. His assistance has been instrumental to me achieving my education and research goals. I would also like to thank Mr. David Levine for all his help navigating the academic world and the many interesting discussions over the years. I would like to express my appreciation to Jiang Ming for taking time to serve on my thesis committee. Finally, I would like to thank Dr. Bahram Khalili for his help and support even before I joined UTA not to mention for introducing me to Dr. Wright.

November 18, 2016

ABSTRACT

EVALUATION OF HTML TAG SUSCEPTIBILITY TO STATISTICAL FINGERPRINTING FOR USE IN CENSORSHIP EVASION

Kelly Scott French, M.Sc.

The University of Texas at Arlington, 2016

Supervising Professor: Matthew Wright

The ability to speak freely has always been a source of conflict between rulers and the people over which they exert power. This conflict usually takes the form of State-sponsored censorship with occasional instances of commercial efforts typically to silence criticism or squelch dissent, and people's efforts to evade such censorship. This is even more so evident in the current environment with its ever-growing number of communication technologies and platforms available to individuals around the world. If the face of efforts to control communication before it is posted or to prevent the discovery of information that exists outside of the control of the authorities, users attempt to slip their messages past the censor's gaze by using keyword replacement. These methods are effective but only as long as those synonyms are not identified. Once the new usage is discovered it is a simple matter to add the new term to the list of black-listed words. While various methods can be used to create mappings between blocked words and their replacements, the difficulty is doing so in a way that makes it clear to a human reader how to perform the mapping in reverse while maintaining readability but without attracting undue attention from systems enforcing the censor's rules and policies. One technique, presented in a related article, considers the use of HTML tags as way to provide a such a replacement method. By using HTML tags related to how text is displayed on the page in can both indicate that the replacement is happening and also provide a legend for mapping the term in the page to one intended by the author. It is a given that a human reader will easily detect this scheme. If a malicious reader is shown the page generated using this method the attempt at evading the censor's rules will be obvious. A potential weakness in this approach is if the tool that generates the replacement uses a small set of HTML tags to effect the censorship evasion but in doing so changes the frequency of those tags appear on the page so that the page stands out and can be flagged by software algorithms for human examination. In this paper we examine the feasibility of using tag frequency as a way to distinguish blog posts needing more attention, examining the means of data collection, the scale of processing required, and the quality of the resulting analysis for detecting deviation from average tag-usage patterns of pages.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Motivation

As the Internet becomes more and more the center for communications it is also the target of efforts by nations who want to control how it is used and the content that is available[2][3][4][5]. There are large number of different topics that this type of control may be appropriate, like state secrets or pornography[6], but there are some topics where that same control is used to repress more social avenues of expression from dissent[7][8][9] to cultural[10][11]. In those cases authors seek a way to speak out without being silenced merely because someone in power decided that discussion of a certain topic, event, or person is off-limits[12].

1.2 Summary of the problem

There has been a tool proposal that seeks to provide a solution for the problem[13] of censorship based on keyword filtering. The replacement method in that paper would be easily detected by a malicious reader. The assumptions that a user needs to make are that the consequences for being caught are low enough, like the post simply being deleted, or that the chances of being detected by an automated scan is low enough, hopefully both. One proposed weakness of that approach is that the HTML tags it uses to mask keywords might influence the distribution of tags in such a way as to be detectable by some censoring authority, essentially that the page appears abnormal in some way that is detectable by some algorithm.

1.3 Challenges

There are several challenges to determining whether HTML tags are susceptible to statistical fingerprinting.

1. Can we identify a corpus of data that is representative of the texts that might use the tool?
2. Can we process enough data to provide a baseline against which to detect the use of the censorship evasion tool?
3. How do you store the processed data to enable providing HTML Tag frequency statistics?

1.4 Contribution

The contribution of this research is the collection and processing of a representative sample of content in an efficient manner in order to determine a baseline of HTML tag frequencies. In addition, we analyze the effort needed to produce statistics on HTML tag frequencies and how practical it would be to use those statistics for use in detecting pages using a keyword replacement technique attempting to avoid censorship.

1.5 Organization of the thesis

Chapter 2 provides the background and talks about the related works, Chapter 3 is where the model and goals are presented, Chapter 4 describes the experimental design, and Chapter 5 lists the results and various statistics. Furthermore, Chapter 6 discusses the conclusions and the potential future work.

CHAPTER 2

Background and Related Work

2.1 Background

Censorship on the Internet has become a sort of arms race between states that want to control the posting of content they find to be objectionable whether because it is embarrassing, vulgar, or potentially seditious and individuals or organizations that want to access information unfiltered by a censor, publish facts that may contradict a party-line, or do either of those things yet remain anonymous[14]. The list of the different types of information states want to restrict depends highly on the goals of each state[15], some want to restrict access to the outside world and so use crude but effective mechanisms like blocking internet access almost completely (North Korea). Some states like Iran allow general access to the Internet but will completely block social network sites like Facebook[16] or Twitter[11][15] or even criminalize user access to those sites[6]. While that approach is somewhat heavy-handed it has the advantage of being easy to enforce. For a country like China, which has a huge population and wishes to use the Internet to power commercial growth, it can be much harder to arrange for access for innocuous material while preventing access to or distribution of content deemed worthy of being censored[17]. The reasons any given piece of content could come under the censor's eye are very broad, so instead of attempting to detail the different categories, this paper labels such content as 'contraband' - meaning any material that gets censored without regard to the subject matter area it might belong or the mechanism used to perform the censorship. Even when a country's stated motivation for applying censorship is to secure their network, it has been shown that countries that use strong censorship do not have any greater network security than countries who only use minimal censorship[18].

2.2 Censorship types

There are multiple ways Censoring Authority or CA can approach censorship, acting directly on the communication request, interfering with the capabilities of the network so requested communications fail, and exerting social, legal, or political pressure on communication nodes so those nodes enact rules set forth by the CA. In addition to the different techniques a CA can implement censorship, the timing of the censorship can be used to define categories and group similar techniques together. This paper categorizes these mechanisms into pre-publish, search manipulation, and post-publish to reflect the different approaches needed for each of them them and focus on post-publish censorship.

2.2.1 Direct Interference

We define direct interference as being when a censoring authority manipulates a connection based upon data it contains which typically means only acting passively on traffic requests but can include when the CA is the source of the connection. One of the earliest methods censors employed to block access to content was by not only blocking connections based on keywords in the URL, including the host name, or deep-packet inspection[3] which allowed restrictions to specific sites without blocking whole ranges of IP addresses. The most well-known example of this is the so-called 'Great Firewall of China', or GFW[3][19][5][20], but even it changes over time and those changes can be mapped[21][15][20] in order to better understand how it functions and avenues of potential circumvention. The GFW has been a passive actor, only seeking to block suspect traffic and never initiating an aggressive posture towards the wider Internet. More recently a new tool called 'The Great Cannon' has been a tool used to attack servers that host Greatfire.org in 2015[22]. The infrastructure used by the Great Cannon may have a connection to the GFW[9]. A recent study of censorship deployment has shown that there is a relationship between the level of censorship adopted by a country and the topology of their network[23] which makes for an unexpected influence requiring more study.

2.2.2 Indirect Interference

Censoring Authorities may want to disrupt communication with a target server or host without regard to any specific property of the request other than where the content resides[9]. Because this approach has such a broad focus it normally has a large amount of collateral damage. For the purposes of this paper, any traffic that is interrupted or compromised and was not the focus of the censor has been indirectly affected. These methods are not as common and may only be temporary.

An example of indirect interference is DNS poisoning attacks[24][4][3] which are crude but effective ways of preventing access to pre-existing content that is hosted on a server that is outside the control of the censor.

2.2.3 Social or Legal Pressure

Search is a logical place for a censor to exert their influence since it has become the dominant discovery tool for content across the Internet. When a search engine operates within the borders of a censoring state, it can be required to apply filters for the set of keywords corresponding to whatever prohibited terms the censor deems necessary. In the case of Google, it decided to stop operating in China rather than filter search results according the demands of Chinese censors[25]. Another aspect of censorship is the detection of contraband content inside the censors network, such as a web article or blog post, that slipped past the various pre-post mechanisms. Flagging such content is a different problem and as such has seen several different mechanisms implemented to catch such postings. In ”Every Rose Has Its Thorns”[12] it is shown that social media platforms in China use a form of self-censorship as a way to prevent government interference with the platform such as heavy fines, a shutdown order or even criminal charges. These platforms can perform these types of keyword filtering before allowing the content to be published and available to the general Internet, so we consider this as pre-publish censorship. The advantages of this are fairly obvious in that the censoring authority does not have to devote resources to the detection and enforcement of the rules, effectively distributing the work to any platform that wants to host content within the jurisdiction of the CA[12].

2.3 Censorship Circumvention

There are many challenges involved in either posting or retrieving contraband since it depends on the security needs of the user and how much risk they are willing to take relative to the expected reaction of the CA, not to mention the known insecurities of the existing tools[26]..

2.3.1 Hiding Identity Through Anonymity

Providing services that allow for anonymous communication is problematic in multiple ways which range from the technical to the economic. Designing a system that can claim enough cryptographic security to protect a users anonymity is not simple[27] and even when those services exist they can still be subject to attacks on their infrastructure like distributed denial of service attacks[9] or political pressure to shutdown a targeted server.

2.3.2 Covert Channels

In order to hide the identity of the server being communicated with, The Onion Router or Tor[27] attempts to provide anonymous communication channels[28] in an untrusted environment where participating nodes may be malicious, using techniques that are conducive to interactive sessions. Since there are legitimate uses for secure connections such as over SSL and HTTPS, censors initially had a choice of blocking all secure connections or tolerating connections to Tor. Censoring countries are constantly changing the techniques used to detect and shutdown access to Tor[19][29]. Even though it is susceptible to traffic analysis and other attacks it is still widely used since it is relatively easy to create new entry nodes[30] and middle nodes which make it harder for a state actor to act as a man-in-the-middle.

Redirection is another approach used to hide the target of communication including at the router level, the DNS request[31], and potentially the protocol level. These systems[32][33] provide a way to connect surreptitiously by inserting special routers into the network that can identify a token which indicates a covert request for content that has been blocked. There are many obstacles[34][35] to implementing such a system.due to its need for widely distributed routers.

2.3.3 Keeping Communication Channels Open

States are not limited to censoring content only before it is published. Using automated tools like search engines to locate offending content makes the task of avoiding the censor's gaze include that of making the verboten post blend in with innocent content[36] since any given reader may be an agent for the censor. Simple keyword replacement schemes are usually the first thing users try, basically creating new euphemisms for banned words or phrases. This usually only works in the short term because once a replacement word is added to the black-list another term is invented to take its place.

For more involved mechanisms to provide keyword replacement such as Collage[37] the idea is to spread the content out over a large enough set of servers that make it more impractical to totally prevent complete coverage without banning large blocks of IP addresses. Collage requires coordination between the various servers hosting the content not to mention the out-of-band communication to match the coordination and needs modified browsers to provide decoding on the client-end.

In the face of an adversary who wants to selectively censor material they consider objectionable but doesn't want to disrupt other content it may be possible to take advantage of that[38][39] by providing posts that could trigger the censor in such a manner so that they are tied to innocuous articles[40] and thus attempt to force the censoring authority into having to decide to block the whole server instead of just the sensitive parts. In this way the author is using the innocent content as a sort of 'cover' for their communications. The model for this paper will extend on this concept.

2.4 Related Works

2.4.1 StegoTorus

Since Tor provides connections over an untrusted network it is vulnerable to traffic analysis[27]. Tools have been created to obscure the patterns of the data so as to foil such analysis. StegoTorus[41] describes a way of wrapping the Tor connection in such a way as to hide the fact that the connection is using Tor.

2.4.2 Message In A Bottle

Message in a Bottle[42] considers how to provide a covert communication channel where the sender of the message *Alice* wants to communicate with the receiver of the message *Bob*. Alice must know something about Bob in order to coordinate communications but that very bit of data regarding Bob can be discovered by the censor as well so they describe a way to establish a link without the need for a rendezvous point.

2.4.3 Economics of censorship resistance

A more recent aspect of censorship resistance is to take into account the costs to both the author and the censor in their efforts to disrupt or overcome their respective adversary[43]. One aspect of this is the practical considerations of attempting to filter the traffic of 1.3 billion individuals constantly in the face of ever-changing political sensitivities and priorities.

2.4.4 GhostPost: Seamless Restoration of Censored Social Media Posts

A more direct challenge to not just the policies of censorship but on the resources deployed to enforce those policies. A new tool described in 'GhostPost: Seamless Restoration of Censored Social Media Posts'[17] does not avoid censorship but rather spreads the prohibited material around to different hosts willing to repost the material and thus engage in a strategy of outlasting the censor by overwhelming the CA's ability to constantly scan a site's entire content in order to find and remove the restored posts. This does rely on having hosts that are willing to take on risk in the name of censorship resistance, but there are mechanisms included to both decrease the chance of subsequent review by the censor and increasing trust among the users, specifically hiding restore posts from anyone not in the trusted list of users and implementing a reputation system respectively.

2.4.5 A Framework for the Game-theoretic Analysis of Censorship

In A Framework for the Game-theoretic Analysis of Censorship Resistance[44] the authors use game theory to help evaluate how censorship resistant systems can optimize their strategies for responding to the censorship apparatus being resisted. By evaluating the behavior of both parties it can be determined if the censor is acting rationally in terms of resources deployed versus success rate which can be used by the CRS to be more effective.

2.4.6 Matryoshka: Hiding Secret Communication in Plain Sight

Much like Edgar Allen Poe's 'The Purloined Letter' hiding information in public view can be very effective at avoiding unwanted attention. Matryoshka: Hiding Secret Communication in Plain Sight[36] proposes a way to encode the message by applying a linguistic steganography approach. The target message can be hidden inside another message in a manner that allows the author to review the overt text while allowing the hidden text to take advantage of steganography's ability to distribute a message without leaving statistical traces.

2.4.7 CovertCast

As the censor gets better at finding ways to prevent the publication of forbidden material it brings out creative solutions by content authors. In the case of CovertCast[7] they avoid the tools that have been developed to process text by translating web pages into video streams in order to hide the content and attempt to force a decision by the CA around blocking all video streams in order to prevent the use of this solution.

Detecting patterns in text to predict whether a given example has a high probability of belonging to a known corpus of identified material is how online email services were able to quell the tsunami of spam filling peoples in-boxes. This was accomplished by applying a Bayesian approach[45] and is still relevant with email filtering.

In order to communicate using existing content hosting platforms it might be possible to hide sensitive material in plain site by adopting HTML as part of the mechanism used to camouflage a message that contains keywords which would otherwise be flagged for action by a censor[13]. One of the weaknesses theorized about this approach is that the modification of the page with a known set of HTML tags might cause the target page to be easier to be identified by a censor due to the presence of an above-average number of those tags. In order to examine this question, there needs to be a baseline for HTML tag frequency in the type of posts subject to filtering by the CA.

CHAPTER 3

Models and Goals

3.1 Censorship categories

3.1.1 Pre-publish

Censorship is in itself a very broad subject area to discuss, describe, and research. Generally, a censor can attempt to convince those responsible for posting content to self-censor whether that is an author who decides to change wording because they are aware of blacklisted keywords or the operator of a social media platform who implements a moderation system in order to keep from being fined by the government. By applying censorship before the content becomes generally available we call this category 'pre-publish'. This provides a way for the censor to effectively control content hosted within the boundaries of their network.

3.1.2 Search Manipulation

Another approach a censor can take is to influence operators of search engines to modify search results either through matching search terms against a list of keywords or removing links to results that point to hosts that either directly contain blacklisted keywords or more likely to be in a list of blacklisted sites provided by the censoring state. Since this method's goal is to hide the existence of the material to make it more difficult to be found we call this category 'discovery modification'. This provides a way to reduce the likelihood that banned posts will be available to searchers within the censors network and also has the benefit of not needing a large investment in hardware or software by the state and would reduce the administrative efforts considerably by offloading the workload onto the search provider.

3.1.3 Post-publish

The remaining category of techniques is detect an attempt to access verboten material published on servers outside the control of the censoring state, this usually takes the shape of keyword blocking in URLs, domain names, even within the data stream itself, otherwise known as deep-packet inspection[46]. This category is post-publish since it occurs after the content has been published but it still depends upon the location of the server hosting the contraband, essentially whether the content is local, within the CA's network, or external, outside the control of the CA. Normally this applies to content that is hosted outside the censors network but depends on the implementation details; locating this feature on the edge of the network so as to only examine traffic transitioning into or out of the censors network would limit the censors ability to affect content hosted within the censor's network that escaped detection of any pre-publish censorship attempts[3].

3.2 Attacker Model

Once a post has been published, it is important to the author to keep from being detected by the systems that implement the CA's policies. For the purposes of this paper we focus on content that has been successfully published inside the CA's network but might be flagged for censorship by systems that examine posts using programs that browse the material as any normal reader would.

3.2.1 Keyword replacement model

Throughout history people have used a wide variety of replacement schemes to hide messages. The first ciphers were just using a shifted alphabet to come up with a simple letter replacement which while simple to use were also easy for an adversary to decode given enough time. Next came word-replacement utilizing code books which were much more secure still relied on the dictionaries used as the key to be available to both the sender and the receiver without falling into the hands of the adversary. In the digital age computers have proven to be powerful at processing text including pattern matching[47] looking for keywords to filter content. Systems for filtering content based on a set of keywords were a natural place for censors to use as the foundation of their efforts to control communication through their networks[2]. The simplicity and ubiquity of keyword filtering algorithms mean they are also useful when authors want to post content about a topic which is known to be on a blacklist[12]. By replacing a blacklisted word with another word that is not on the blacklist the author can bypass the filters as long as the replacement is easy for a human reader to identify and revert. The 'Grass Mud Horse' dictionary is an example of this that has become very popular way for authors circumvent the restrictions on referencing sensitive terms, essentially a simple form of substitution cipher. With this in mind, an approach for evading censorship[13] has been proposed that manipulates the message using HTML tags in order to both replace the blacklisted keyword with another term and also provide a legend for the replacement that is maintains a reasonable degree of readability by the reader.

3.2.2 Example

If the tool identified the word 'chinese' for replacement using the Colored Dictionary type, it would mark the keyword with HTML tags to modify the font and or color like so:

<**font color**="red"><**strong**>Chinese</**strong**></**font**>

In fig. 3.1 we show an how this example of the colored dictionary type would be rendered but we did not perform keyword replacement in order to highlight how the keywords are surrounded with select HTML tags by the tool.

3.2.3 Statistical Flagging

One weakness of a substitution cipher is frequency analysis[48] so when there is a tool that modifies the HTML content of a page in a mechanical fashion it is fair to question whether the output of the tool could introduce a pattern[47] that is detectable using frequency analysis. In order to determine the vulnerability of such tools[13] to such analysis we would first need a baseline with which to compare the tools output to see if the introduction of HTML tags significantly change the the tag frequency opening the door for unwanted attention by the CA.

3.2.4 Tag Selection

One of the goals of using HTML tags to provide masking for blacklisted terms is to do so in a way that does not inhibit readability[13]. The available choices were to use text-only replacement, with variations for keeping the text plain vs colored, or to include images of the letters or words to increase the difficulty of a censor to perform frequency analysis. The results of [13] showed that both text replacement and image replacement were technically feasible but the image replacement technique was considered to have better readability although not by a significant amount.

For this paper the use of images is out of our scope mainly because even using images to provide the final form of the transformed message still requires the introduction of HTML tags that wrap the message and it is the inclusion of these tags, and their potential to change the tag frequency of the web page, that are the focus of this work.

3.3 Assumptions

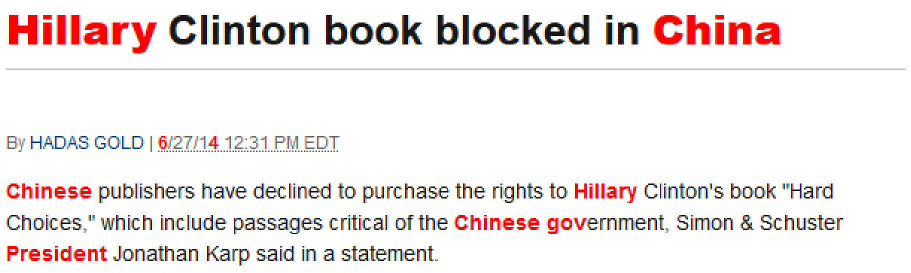
This theorized vulnerability assumes that it might be possible to detect changes in the rate a combination of tags appear on a blog and use that to mark a post for further attention by a human censor. If examined by a human, they would easily detect the attempted evasion which might mean more serious consequences for the author. In order to be flagged for further attention there would need to be a way to determining a baseline tag frequency in practical terms of the resources deployed by the detection effort.

If each company hosting content is left to it's own devices to implement censorship mechanisms, rules, keywords[12], etc. then there is likely no central data collection mechanism outside known censorship points like the great firewall and others that already filter traffic in real-time.

To attempt to detect the technique of implementing an arbitary code language by replacing keywords with images [13] based on the increased use of the HTML tags used to embed the replacement images implies several things; that there exists a baseline tag frequency, that said frequency stays within a narrow enough range to be useful, that the embedding method is static, that tag frequency within single pages stay close to the baseline with no wide swing in frequency between pages, that it is feasible to collect tag frequency stats for all pages or a baseline; that it is feasible to collect frequency stats based on any combination of tag hierarchy (font tags containing strong tags); that said combo identification is robust (limitations of xpath / xquery to describe all possible parent/child relationships between target tags).

3.4 Goals

Goal is to capture statistics for pages containing both image and font related tags along with the performance information including storage requirements, processing time needed, and any processing challenges to assess the feasibility for using tag frequency in locating attempts at censorship evasion. Some tags are definitely related to the font tag but have been deprecated or are not supported in HTML5.Tags under consideration for data gathering are <B>, <BIG>, <EM>, <STRONG>, <SMALL>, <I>, <MARK>, <TT>, and <U>. See table A.1 in appendix A for full list of tags.



Hillary Clinton book blocked in China

By HADAS GOLD | 6/27/14 12:31 PM EDT

Chinese publishers have declined to purchase the rights to Hillary Clinton's book "Hard Choices," which include passages critical of the Chinese government, Simon & Schuster President Jonathan Karp said in a statement.

Figure 3.1. Colored Dictionary Example.

CHAPTER 4

Experimental Design

4.1 Data Collection Methods

4.1.1 Screaming Frog

Initially we attempted to crawl the internet ourselves so the data would be as fresh as possible but we encountered several challenges. Crawling the Internet is a complex, resource-intensive task. One approach we considered was to use a web-crawling tool named 'Screaming Frog' because it already handles the link-spidering and HTML parsing but it requires a starting list of links that it can visit. Finding a relatively large list of links is not very straightforward. Even a simplistic technique of starting with a known blog on the popular blogging host Blogspot and then using it's built-in navigation links to visit a series of pages did not result in usable data because either the crawl rules would restrict the depth of link harvesting from visited pages or some blog would not enable the header component containing the blog-navigation links which would restrict the linear link harvesting. Another limitation of Screaming Frog is it uses a set of rules based on regular expressions to extract specific information from crawled pages and then would pipe the output to essentially a log file. This hampered the ability to look for all HTML tags in a page because each tag would require it's own rule and regular expressions are notoriously bad at handling HTML parsing[49].

4.1.2 Spinn3r

Crawling the web ourselves would be fine except it would have been hard to scale up easily. Once we were made aware of an existing source of HTML content from the Internet, we evaluated the new source provided by Spinn3r to attendees of the 2011 ICWSM[1] and decided to focus on it. The new data source was relatively recent, from 2011, and was considerably larger as shown in table 4.1. The total size of the data-set is 2TB compressed and the subset containing weblogs has a size of 1.46TB compressed. For comparison, the archive for the weblogs on Jan 24th has a compressed size of 53.8 GB and an uncompressed size of 62.9 GB. The sheer amount of data usually allows for a more comprehensive study but the size of that same dataset became a challenge itself as managing large data-sets increases all aspects of data processing, from the time it takes to acquire , parse, and process, to the space needed to store the raw data and the structures needed for processing and reporting. Spinn3r provided a library written in Java to retrieve structured information from the streams that store the unstructured data captured from the Internet.

Table 4.1. ICWSM 2011 Dataset[1]

Content Type	**# of Elements**
Weblog	133683918
Mainstream News	14744094
Classified	517373
Forum	5734378
Review	32773
Social Media	231861650
Memetracker	2473
Total	386576659

4.2 Available Content

4.2.1 Corpus Description

The data examined for this paper were Weblogs written in English and posted over a week in January of 2011, January 13th through January 17th with some statistics about data size through the 24th. The Spinn3r data was divided into several categories (Social Media, Weblog, Other). The 'Weblog' category was chosen because it most closely matched the use model for the type of activity that could involve being flagged for post-publish censorship. In addition the content provided from the stream is the base HTML file retrieved from the server and has not been expanded. Any files intended to be included by the client browser are not included in the content which means only the inline CSS and scripting is visible to the parsing subsystem. It is possible that there are included CSS or scripts that define transformations used to modify the font but to check for that would require a much more robust HTML support such as the back-end of a browser's rendering engine since such processing happens in order to determine how to display the page content.

4.2.2 HTML

From a previous project we had PHP code for manipulating web pages where a user would a provide a URL, the code would retrieve the HTML for the page and then parse it for specific information using a proper HTML parser, in this case we used the JSoup parser written in Java. This code was modified to accept the raw HTML directly, parse the HTML looking for all HTML tags, and keep a hash table of all the tags it found along with a count of the number of times the tag was found. This information was stored by connecting to a MySQL instance and issuing an SQL Insert statement into tables to capture several aspects of the data, mostly in two parts; information on the source of the data so we could group data by the day it came from, and the breakdown of a given page into a list of tags and a number of times it is used on that page.

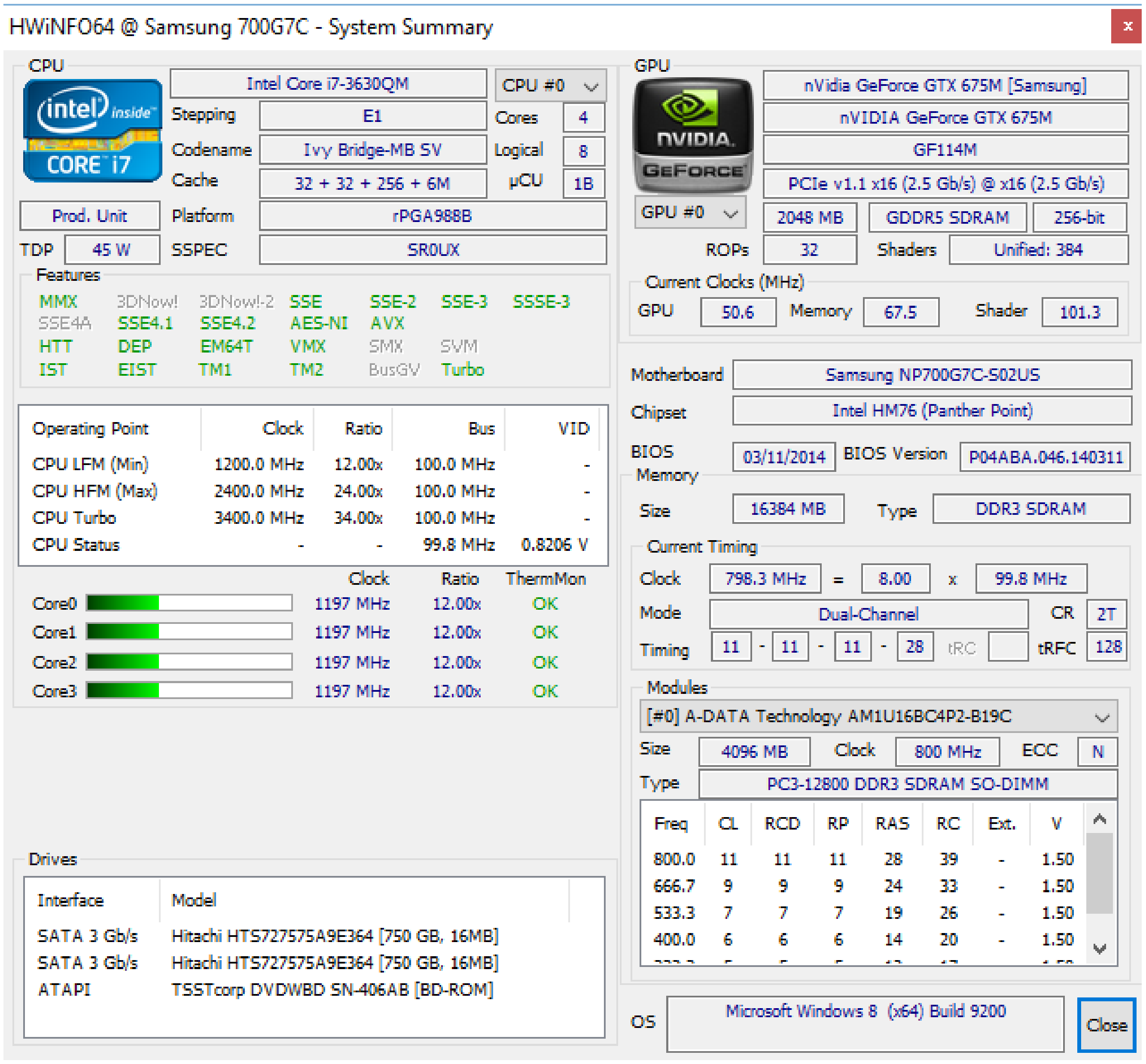
4.2.3 Language Selection

English was selected for the initial study with plans to expand to other languages if we found gaps in the language distribution. Also, it made it simpler to design all the components without worrying about languages that used different codepage or required double-byte encoding. As shown later in table 5.2 using English encompasses a large percentage of the languages in the corpus. Covering more languages and / or types of posts would increase the storage / processing requirements, so we have a high degree of confidence that the results of this paper are conservative estimates compared to the contents of the entire Internet.

4.3 Processing

4.3.1 Hardware

The computer used to process the data has the following hardware: Processor Intel(R) Core(TM) i7-3630QM CPU @ 2.40GHz, 2401 Mhz, 4 Core(s), 8 Logical Processor(s) with 16GB of memory, 2 750GB 7200RPM hard drives, see fig. 4.1. The impact of this there was plenty of CPU horsepower available for processing and by bringing all the data to a single machine it removed the network as a source of bottlenecks. The main limitation for processing and analytics came down to disk I/O rates, the size of the data grew with every day added to the tag count table that any sort of data manipulation required ever-increasing amounts of data and thus the dependency on data throughput.



HWiNFO64 @ Samsung 700G7C - System Summary

CPU

	Intel Core i7-3630QM	CPU #0	
Stepping	E1	Cores	4
Codename	Ivy Bridge-MB SV	Logical	8
Cache	32 + 32 + 256 + 6M	μCU	1B
Prod. Unit			
Platform	rPGA988B		
TDP	45 W		
SSPEC	SR0UX		

Features

MMX 3DNow! 3DNow!-2 SSE SSE-2 SSE-3 SSSE-3
SSE4A SSE4.1 SSE4.2 AES-NI AVX
HTT DEP EM64T VMX SMX SVM
IST EIST TM1 TM2 BusGV Turbo

Operating Point	Clock	Ratio	Bus	VID
CPU LFM (Min)	1200.0 MHz	12.00x	100.0 MHz	-
CPU HFM (Max)	2400.0 MHz	24.00x	100.0 MHz	-
CPU Turbo	3400.0 MHz	34.00x	100.0 MHz	-
CPU Status	-	-	99.8 MHz	0.8206 V

	Clock	Ratio	ThermMon
Core0	1197 MHz	12.00x	OK
Core1	1197 MHz	12.00x	OK
Core2	1197 MHz	12.00x	OK
Core3	1197 MHz	12.00x	OK

Drives

Interface	Model
SATA 3 Gb/s	Hitachi HTS727575A9E364 [750 GB, 16MB]
SATA 3 Gb/s	Hitachi HTS727575A9E364 [750 GB, 16MB]
ATAPI	TSSTcorp DVDWBD SN-406AB [BD-ROM]

GPU

nVidia GeForce GTX 675M [Samsung]
nVIDIA GeForce GTX 675M
GF114M
PCIe v1.1 x16 (2.5 Gb/s) @ x16 (2.5 Gb/s)
GPU #0 | 2048 MB | GDDR5 SDRAM | 256-bit
ROPs 32 | Shaders Unified: 384

Current Clocks (MHz)

GPU 50.6 | Memory 67.5 | Shader 101.3

Motherboard	Samsung NP700G7C-S02US
Chipset	Intel HM76 (Panther Point)
BIOS	03/11/2014
BIOS Version	P04ABA.046.140311

Memory

Size 16384 MB | Type DDR3 SDRAM

Current Timing

Clock 798.3 MHz = 8.00 x 99.8 MHz
Mode Dual-Channel | CR 2T
Timing 11 - 11 - 11 - 28 tRC | tRFC 128

Modules

[#0] A-DATA Technology AM1U16BC4P2-B19C

Size 4096 MB | Clock 800 MHz | ECC N
Type PC3-12800 DDR3 SDRAM SO-DIMM

Freq	CL	RCD	RP	RAS	RC	Ext.	V
800.0	11	11	11	28	39	-	1.50
666.7	9	9	9	24	33	-	1.50
533.3	7	7	7	19	26	-	1.50
400.0	6	6	6	14	20	-	1.50

OS Microsoft Windows 8 (x64) Build 9200

Close

Figure 4.1. Hardware capabilities.

4.3.2 MySql

The relational database engine MySQL was chosen to house the processed data as it enables the capability to produce many different breakdowns of the data. MySQL was installed on my laptop which has above-average hardware but its performance comes down to whichever component is the slowest and thus becomes the bottleneck, in this case the bottleneck was the hard drive I/O bandwidth.

4.4 Data Model

4.4.1 Input Data

Spinn3r stores data in a compressed format and provides a library for extracting pages from the stream. The program that processed the streams was given the date, the datatype (weblogs), and the language as arguments. The code would then enumerate all the files that match the provided parameters, open each stream and then iterate over each stream. A stream contained up to 200 pages. From each page, the host and page would be stored separately in order to group all tags on a page together in the database.

The focus of this work is the counting the number of times an HTML tag appears in a page, so we needed a table to store tag occurrences and a table to help organize those entries. The table 'tagfile' in listing 4.1 tracks the arguments used along with file corresponding with the input stream.

Listing 4.1. Tracking Input Pages

```
CREATE TABLE `tagfile` (
  `idtagfile` int(11) NOT NULL AUTO_INCREMENT,
  `type` varchar(45) DEFAULT NULL COMMENT 'Spinn3r␣datatype,␣mostly␣WEBLOG',
  `language` varchar(20) DEFAULT NULL,
  `month` varchar(10) DEFAULT NULL COMMENT 'the␣month␣value␣from␣the␣Spinn3r␣structure',
  `day` varchar(10) DEFAULT NULL COMMENT 'the day value from the Spinn3r structure',
  `path` varchar(200) DEFAULT NULL COMMENT 'the path argument from processing a batch of Spinn3r 
      data',
  `filename` varchar(300) DEFAULT NULL COMMENT 'name of the file from the Spinn3r structure',
  PRIMARY KEY (`idtagfile`),
  KEY `DateIdx` (`month`,`day`) COMMENT 'Speed queries focused on month and day'
) ENGINE=InnoDB AUTO_INCREMENT=106959 DEFAULT CHARSET=utf8 COMMENT='Pages are grouped by file in 
    the dataset, this table contains the details of the file and enbles the association beteen a 
    page and the file from which it was processed.';
```

This simplified deleting and reloading data as the processing code was updated during development. The table 'tagcount' in listing 4.2 stores the tag name and count of occurrences on the page while linking back to the page and input file.

Listing 4.2. Tracking Parsed Tags

```
CREATE TABLE `tagcount` (
  `host` varchar(200) NOT NULL,
  `page` varchar(600) NOT NULL,
  `retrieved` varchar(20) NOT NULL,
  `tagname` varchar(40) NOT NULL,
  `tagcount` bigint(20) NOT NULL,
  `fileID` int(11) DEFAULT NULL,
  KEY `FileIdx` (`fileID`),
  KEY `idx_tagcount_page` (`page`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

During our work we found that using a single table for multiple days worth of tag statistics was useful but as more data was stored the query performance degraded quickly. For this reason we began to create tables for each day of input data. As we narrowed in on the types of statistics we wanted we created three tables for collecting the number of pages that contained either of two tags and both tags together. This can be seen in listing 4.3 where we query the list of pages containing tag 'A', store the results in a separate table and repeat it with tag 'B'. That allows a join between those temp tables that give the count of pages for a specific day containing both tags. This approach has a small advantage in that if the first tag is help constant the script can be run to only clear and update the temp table for the second tag, saving time when looking at multiple tag combos that have one tag in common. The other advantage to this approach is the script can be parameterized so that the script is executed multiple times, each time providing a set of tags and tag combinations with no human intervention.

Listing 4.3. Gather Statistics On Tag Combinations in Pages

```

--
-- **** gather stats for how many pages have a combination of two tags
--

set @tag1 = 'font';
set @tag2 = 'img';
SET @bothtags=CONCAT(@tag1, '-' , @tag2);

set @qmonth = '01';
set @qday = '13';

truncate pages_tag1;
truncate pages_tag2;

insert into pages_tag1 (select distinct tc.page
        from tagcount tc
    inner join tagfile tf on tc.fileid = tf.idtagfile and tf.month = @qmonth and tf.day = @qday
    where tc.tagname = @tag1
    );

insert into pages_tag2 (select distinct tc.page
        from tagcount tc
    inner join tagfile tf on tc.fileid = tf.idtagfile and tf.month = @qmonth and tf.day = @qday
    where tc.tagname = @tag2
    );

set @tag1count = (
select count( distinct page)
from pages_tag1);

select @tag1count;

set @tag2count = (
select count( distinct page)
from pages_tag2);

```

```
select @tag2count;

set @bothtagscount = (
select count(distinct pt1.page)
from pages_tag1 pt1
inner join pages_tag2 pt2 on pt1.page = pt2.page);

insert into tagstats values (@qmonth, @qday, @tag1, @tag1count);

insert into tagstats values (@qmonth, @qday, @tag2, @tag2count);

insert into tagstats values (@qmonth, @qday, @bothtags, @bothtagscount);
```

CHAPTER 5

Results

The processing of the data was a challenge in several ways, handling the compressed data, processing the data streams into more usable form, and doing analyis on the processed data.

5.1 Data

5.1.1 Archive

In table 4.1 it is shown the size of the entire ICWSM2011 dataset but in table 5.1 it is shown how much data is involved in the date range studied in this paper, storage in GB for the uncompressed data, number of Spinn3r streams, total number of tags present in all pages that day, and the number of pages. A stream in the dataset is a collection of 200 pages of that day's corpus for the category, weblogs in this case.

Table 5.1. Data Corpus Size

Date	Storage	Streams	Tags	Pages
2011 01 13	4.19 GB	1445	189,312,857	205011
2011 01 14	24.2 GB	8684	1049930682	1210359
2011 01 15	20.9 GB	7501	910488269	1083455
2011 01 16	19.8 GB	7196	866109739	1056493
2011 01 17	30.4 GB	10425	1341451543	2084673

5.1.2 Language

Initially the entire archive would be extracted which included files for all languages crawled, but in order to simplify the processing requirements it was decided to only process files representing weblogs written in the English language. This is borne out by comparing the ratio of the size of the different languages content. In table 5.2 it is clear that almost half of all the weblog content is in English and the percentage of Chinese, obtained from the content marked with language code 'zh', is extremely small in comparison. Why this would be is unknown, our guess is either most content hosting site use engines that default to English, weblogs hosted in China are not visible to the crawling process, or the Spinn3r engine had problems processing that content. This last one is unlikely considering how many languages were present including Chinese. The estimated data was due to the uncompressed data for all languages except two for the affected day being deleted to recover hard disk space, using a simple percentage that was close to the English ratio for the other days rounded up, allowed me to use the actual space used by the English content for that day to then roughly derive the size of the entire corpus.

5.1.3 Database Storage

The main table used to store the tag counts for data captured for the days Jan 13th 2011 through Jan 17th 2011 ended up with 201,311,230 rows for a total table size of 32.GB for an average row length of 161 bytes. An index was defined to help with data analysis and that added another 4GB to the storage consumed by the table.

5.2 Processing

There was considerable effort expended in all stages of data handling due to the size of even a single day from the corpus. The selected data for a day had to be

Table 5.2. Language comparison

day	uncompressed	English	% English	Chinese	% Chinese
13-Jan	9,076,436,059	4,501,390,667	50%	36,490	0.0004%
14-Jan	53,808,564,035	26,080,194,474	48%	527,308	0.0010%
15-Jan	49,356,836,228	22,469,828,558	46%	906,102	0.0018%
16-Jan	49,341,218,728	21,355,353,840	43%	600,207	0.0012%
17-Jan	72,683,007,594	32,661,903,106	45%	737,122	0.0010%
18-Jan	52,218,099,489	25,424,450,691	49%	358,618	0.0007%
19-Jan	65,593,299,374	31,270,968,621	48%	731,229	0.0011%
20-Jan	67,035,070,435	30,837,263,676	46%	315,087	0.0005%
21-Jan	53,702,303,437	25,741,490,231	48%	1,068,727	0.0020%
22-Jan	46,075,106,534	23,037,553,267	50%	1,288,649	0.0028%
23-Jan	48,916,232,309	21,686,488,651	44%	697,213	0.0014%

extracted from a compressed archive into streams on the file system that could then be parsed for tag information that would then be stored in a relational database for later analysis using SQL queries.

5.2.1 Archive Extraction

Considering each day of weblog data would require almost 50GB of uncompressed space it not possible to extract the entire dataset of over 30 days of posts in more than 20 languages so the procedure was to extract a single day for processing at a time. Initially we were going to process all languages but once it was discovered that English was the predominant language the other languages were no longer considered for processing. In practice this meant that the extraction could be simplified by only selecting English and Chinese streams since we still wanted to gauge how applicable the results were for Chinese weblogs based on the relative sizes. Extracting a single day of weblogs including all languages took roughly an hour each, mainly due to the archives residing on an external 3TB hard drive which utilized the USB interface for file access. The file extraction targeted one of the two local hard drives in order to provide for better throughput by using different spindles for read and write while leaving the system drive for the operating system's use. This turned out to be a bottleneck because we found that the second hard drive would be overloaded with write requests and the USB interface having spare capacity. Once this was determined we experimented with extracting the archives back to the 1TB external hard drive but connecting it using a USB 3.0 port; this resulted in a significant decrease in the time it took to extract a day's worth of data from the archived form.

5.2.2 Data Stream Parsing and Storage

Processing a single day of the Spinn3r data took at least 6 hours and sometimes close to 9 hours depending on how many streams were captured for that day. The CPU utilization was not significant and reflected the single-threaded nature of the program doing the extraction from the streams. There are opportunities for implementing a multi-threaded process using a manager-worker thread arrangement but this was not done for this project. Each day processed produced a list of distinct domains and how many times they showed up in the data, see table 5.3. Note, no effort was made to categorize the content of these domains or pages.

Processing one day of data involved selecting the day, source (weblog), and language for processing. The program would then enumerate all the streams for the selected day / source / language and iterate over each stream to extract each page. From each page the URL was parsed to get the hosting domain and the page portions of the URL in order to organize the tag data by page; this data stored in a table of its own to facilitate later data analysis. Once the host and page was found the page was then parsed for all HTML tags, only the opening tag was counted, using a hash table that stored the actual number of occurrences for each tag. Once all tags for a page had been found, the tag hash table was then used to create SQL insert

Table 5.3. Distinct Domains on Jan 13 2011

Domain	1/13/2011 Occurrences
http://w-47.com	8998
http://mouthshut.com	1738
http://blog.fc2.com	625
http://photopoints.com	509
http://photopoints.com:80	509
http://ehow.com	474
http://utube.smashits.com	468
http://feedagg.com	423
http://mississippistate.scout.com	420
http://sea.scout.com	416
http://overseas.vivastreet.co.uk	382
http://tvtropes.org/pmwiki/changes.php	354
http://tvtropes.org	354
http://tvtropes.org/pmwiki/posts.php?discussion=v2xls928cut2ghikfc993pjg	354
http://downarchive.com	344
http://tastro.org	336
http://bittorrent.am	323
http://brooklynpaper.com	309
http://twittorati.com	300
http://globalquad.com	300

statements into a table reserved for tag counts. This table became enormous when trying to store multiple days worth of tag counts so at some point a separate table was created for each day and the program was modified to point to the table that corresponded to the selected day see table 5.4.

5.2.3 Statistics

Once all pages for a day were processed it was possible to calculate how many pages there were, how many tags were in all pages, and the breakdown of both the total occurrences of each tag and how many times each tag appeared on a percentage basis compared to the total number of tags that day, see fig. 5.1. Initially we

Table 5.4. Tag Count DB Storage by day

Date	Rows	Data (MB)	Index (MB)	Storage (MB)
18-Jan	47688856	6,700.00	836.20	7,600.00
19-Jan	58065236	8,200.00	1,013.90	9,200.00
20-Jan	57183202	8,200.00	1,000.80	9,200.00
21-Jan	47334774	6,800.00	831.10	7,600.00
22-Jan	42359479	6,100.00	741.60	6,800.00
23-Jan	40784185	5,800.00	713.10	6,500.00
Totals	293415732	41,800.00	5,136.70	46,900.00
Averages	48902622	6,966.67	856.12	7,816.67

attempted to analyze the tag frequency of single tags over several days to understand the range of values that could be expected.

In table 5.5 we can see that the maximum number of times the 'strong' tag appeared on a single page was 603 times in the data for January 13th while appearing on a page 4,699 times in a page on January 16th.

Table 5.5. STRONG Tag Occurrences per page

Month	Day	Tag	Max	Avg
1	13	strong	603	8.3761
1	14	strong	2438	8.9222
1	15	strong	2439	9.3948
1	16	strong	4699	9.4931
1	17	strong	2439	9.5415

In table 5.6 we can see that the maximum number of times the 'font tag appeared on a single page was 603 times in the data for January 13th while appearing on a page 4,699 time in a page on January 16th.

In addition, it was possible to determine the frequency a given tag appeared in pages based on the total number of occurrences sorted by most used to least used,

Table 5.6. FONT Tag Occurrences per page

Month	Day	Tag	Max	Avg
1	13	font	3730	11.8255
1	14	font	3706	13.0886
1	15	font	6995	11.832
1	16	font	16233	11.0311
1	17	font	6954	13.0349

see table A.2 in appendix A for the complete list of tag occurrences. The question became how many pages contain either of two tags and then how many pages contain both tags. This information was obtained for the Font, Strong, and Img tags and is shown in table 5.7. For combinations of tags we collected the percentages for two combinations, 'Font and Strong', and also 'Font and Img'. It is apparent that for a single tag or pair of tags there is a large distribution in occurrences both of tags individually or in combination with others.

Table 5.7. Pages with Selected Tags

# Month	Day	Pages	Font	% Font	Strong	% Strong	Img	% Img
1	13	205011	42341	20.653	153672	74.958	194247	94.750
1	14	1210359	434272	35.880	970702	80.200	1132222	93.544
1	15	1083455	385985	35.625	833686	76.947	1018187	93.976
1	16	1056493	370273	35.047	785323	74.333	986308	93.357
1	17	1456589	533587	36.633	1136602	78.032	1383516	94.983

Table 5.8. Pages with Selected Tag Combinations

# Month	Day	Pages	Font & Strong	% Font & Strong	Font & Img	% Font & Img
1	13	205011	41402	20.195	42143	20.556
1	14	1210359	167466	13.836	264927	21.888
1	15	1083455	154969	14.303	247638	22.856
1	16	1056493	151845	14.373	242487	22.952
1	17	1456589	202050	13.871	323820	22.231

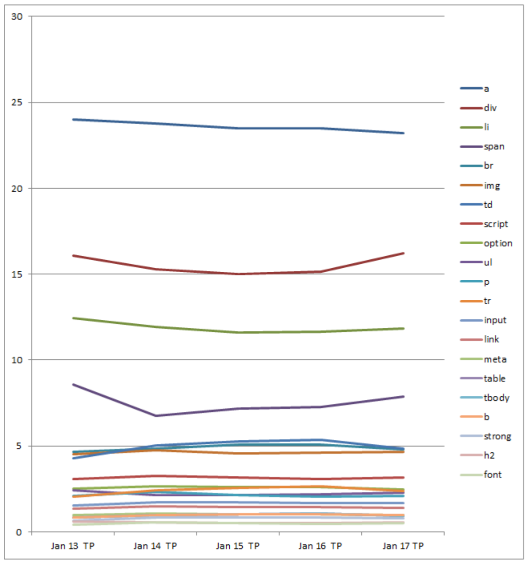


Figure 5.1. Tag Usage Percentage By Day.

CHAPTER 6

Conclusions And Future Work

6.1 Conclusion

A Censoring Authority which desires to detect pre-posted content that should be routed to a human censor for closer study based on the statistical variation of tag usage would be facing an enormous challenge in terms of resources both with storage and processing, not to mention the difficulty in obtaining a practical threshold for tag usage. Individual tag usage have a very large range in terms of the number of times they appear in any given page, one page might use a tag once while another may use that same tag thousands of times. The ratio of tag usage does not stay the same across days so any threshold generated by examining a single day will not be accurate on the next day or any day after. An effort to collect data across many days in order to come to an average usage percentage faces a new problem of data storage where all the statistics needed have to be crawled, processed, and stored during the examined time period before calculating the threshold and even after that is done the threshold is still vulnerable to the shifting usage patterns and so may not be valid as soon as it is found considering how much processing time it takes to the data collected for any significant time period.

How can you update the stats dynamically considering new content is constantly being produced? Do you store all content for a given period for processing later? If you do that, doesn't that mean you have to process all your content twice, once to store it and a second time to process statistics? Does that exclude the 3rd time being the actual publishing of the content? Do you 1) publish, 2) store, 3) process,

4) censor? This means that censored content would be published for some period of time before being detected.

The alternative is to delay the publishing so that periods content is buffered for process like so, 1) store, 2) process, 3) censor, 4) publish. This might reduce the liability of the host service but introduces significant latency between content submission and publishing when the responsibility for censorship has been delegated and distributed[12].

With what we know about the different censorship methods, their distribution across the network, the different focus and granularity at each node, what can we say about the censorship regime as a whole? Each level of enforcement is very different from the rest. Network level methods focus on processing streaming data which limits how the censor can respond, i.e. spoofing a NAK on a socket, intentionally disrupting a DNS result[3], simple keyword detection and URI filtering. Network level sniffing also requires a very large commitment of resources to provide coverage in terms of volume and distribution.

Site-level censors do not have the same level of resources to do comprehensive collection and processing of content so censorship techniques would be similarly scaled back relative to a network-level mechanism unless some manner of coordination as established such s a designated source for keywords which, while possible, also negates some of the advantages of the distributed approach.

As a content author, the inconsistencies of these different levels of censorship can serve as a deterrence, making the experience seem arbitrary and somewhat inscrutable. This works to the advantage of the CA because it makes it more difficult for any single workaround to be effective, it also increases the perception that its only a matter of time before one gets caught. This can influence users perception of increased risk of detection, the result of which can be an increase in the overall deterrence effect of the censorship scheme.

In trying to produce these statistics I have concluded that attempts to censor content based on tag frequency requires too large an investment in resources and is too easily defeated or degraded and thus is ineffective in addition to being highly impractical.

6.2 Future Work

If this experiment were to be repeated we would recommend taking a distributed approach to allow for the collection an processing of much larger date ranges[47], but doing so would still not address the underlying issue regarding the extreme range of valid tag frequencies. While tag frequency alone may not be enough to detect pages using tags to mask the use of keyword replacement, it might be possible to use Bayesian methods much like with spam filters[45]. It might be feasible for a tool such as in [13] to randomly select the embedding technique to further mask it's behavior.

A more accurate gauge of the susceptibility of tag frequency might be to collect statistics on how often certain combinations of tags and attributes occur. This is because there are many instances where font modification happens within the 'STYLE' tag using the appropriate HTML attribute. This would require using more sophisticated methods for data collection. For example XPath or XQuery could be used but it is not clear how practical that approach would be due to the need to define patterns matching essentially arbitrary numbers of combinations of tag nestings and attributes.

A question for the future is if a site (or web host / provider) collected stats only for their content, would detection be easier / harder when the embedding tool can randomly select the embedding snippet? That is, a) are posts on that site similar enough in terms of the HTML they generate, and b)is the scale involved by limiting the statistics collection to a single site small enough to make the effort practical?

APPENDIX A

TAG DATA

Table A.1. The List of Font Related HTML Tags (from w3schools.com)

B	bold
BASEFONT	not usefull because it usually only appears once in a document
BIG	not supported in HTML5
EM	emphasis, usually italics
STRONG	important
SMALL	smaller text
SUB	subscript
SUP	superscript
CODE	a piece of code
SAMP	sample output
KBD	keyboard input
VAR	variable
FONT	not supported in HTML5(?!?)
I	alternate voice or technical term (usually displays as italic)
(INS/DEL)	shows inserted or deleted text
MARK	show text as highlighted
OUTPUT	not supported in Internet Explorer
PRE	pre-formatted text (preserves spaces and line breaks)
Q	short quotation
S	was strikethrough in HTML4.01 but in HTML5 means text that is no longer correct
STRIKE	not supported in HTML5, use ¡s¿ or ¡del¿ instead
TT	originally for teletype but is same as input or code so other tags
U	text that should be stylistically different, usually shown as underlined

Table A.2: Tag Occurrences 2011 - sorted

tagname	Jan 13 Total	Jan 14 Total	Jan 15 Total	Jan 16 Total	Jan 17 Total
a	45447761	249795352	213995053	203311519	311519576
div	30440250	160699022	136740967	131062132	217483483
li	23524882	125365511	105591346	100876864	158664574
span	16241991	70916344	65317987	62879042	105821761

Continued on next page

Table A.2 – continued from previous page

tagname	Jan 13 Total	Jan 14 Total	Jan 15 Total	Jan 16 Total	Jan 17 Total
br	8804688	51029648	46384208	43930625	64533457
img	8574644	49802559	41420078	39955197	62573897
td	8109022	52679246	48123445	46590625	64790389
script	5849548	34210700	28951044	26870849	42745335
option	4759384	27892541	23952562	22698594	33259887
ul	4582286	22402919	19436047	18889856	30927869
p	4011395	24550454	19689724	17704296	28134895
tr	3928576	25275298	23507384	22925212	31745449
input	2900994	18115351	15704015	14619672	22354860
link	2584622	15854271	13208087	12410017	18805540
meta	1850484	11236034	9377744	8956502	13103158
table	1630302	10233420	9308708	9140204	12720557
tbody	1627501	10226834	9302508	9125832	12698967
b	1611324	10538470	9359376	8881861	12900117
strong	1287176	8660835	7832353	7455113	10844835
h2	1174495	5721082	4889309	4538393	7626633
font	784786	5684014	4566958	4084535	6955275
h3	777098	4473587	3763377	3333659	5388462
form	577381	3859909	3248432	2987656	4283387
label	557124	3664188	2920585	2629515	4133075
noscript	409497	2445182	2008900	1891781	2939016
em	389830	2474241	1993512	1851033	2746835

Continued on next page

Table A.2 – continued from previous page

tagname	Jan 13 Total	Jan 14 Total	Jan 15 Total	Jan 16 Total	Jan 17 Total
style	386663	2121025	1857006	1827766	2777763
h4	379750	2137857	1691176	1498499	2500179
dd	377761	1898346	1872609	1850645	2809499
h1	333296	2037593	1810589	1723036	2412594
th	332133	3145730	3965111	1755316	3259929
iframe	315314	1737460	1520858	1427925	2265526
title	295377	1782706	1526149	1457980	2140239
head	288713	1735961	1499697	1438707	2083930
html	288713	1735961	1499697	1438707	2083930
body	288508	1733969	1498347	1437057	2082525
small	277971	2890995	2795406	2618649	3185115
center	274364	1678255	1590941	1509134	2142619
i	265098	1316835	1299171	1288289	1696072
hr	219233	1460286	1272854	1256450	1694274
dt	204002	1063803	1073065	1032516	1649271
dl	188410	981011	959807	942117	1341777
param	183226	992477	866409	789766	1193479
button	151858	1060748	942791	943283	968369
h5	149112	919874	771677	757196	1009714
textarea	129643	823034	725964	660736	1018047
select	109631	696992	605673	559632	887957
ins	95080	551354	692549	852175	908451

Continued on next page

Table A.2 – continued from previous page

tagname	Jan 13 Total	Jan 14 Total	Jan 15 Total	Jan 16 Total	Jan 17 Total
h6	92895	518126	419669	390883	506601
del	92585	551421	690556	836955	884460
ol	90418	568723	535402	505880	665383
fieldset	84763	449711	389537	333173	597737
abbr	80959	417082	388205	387098	565448
sup	74876	292208	320909	302192	365990
cite	64797	388916	324591	271978	398070
blockquote	58511	278435	258496	265643	337359
area	58367	402309	362805	341556	477173
col	48083	245320	272695	293564	330088
object	47965	262842	234045	219311	333209
embed	46177	221633	205733	198282	307268
u	43727	282837	262917	257020	362469
code	34486	93531	75388	66240	86770
header	34350	144586	156524	172572	228857
pre	32002	83244	66937	48125	84251
legend	31583	179410	153729	132306	238977
aside	31480	118053	129931	136387	229113
footer	28286	116755	124152	136396	189248
nav	25741	139384	156028	160203	167432
section	24729	182840	230006	195649	185142
wbr	24722	183857	215947	249661	263445

Continued on next page

Table A.2 – continued from previous page

tagname	Jan 13 Total	Jan 14 Total	Jan 15 Total	Jan 16 Total	Jan 17 Total
thead	23680	123972	135789	124318	173741
colgroup	19161	220102	246004	253645	268152
strike	16414	391021	461036	447534	443005
big	16087	99167	96864	116034	131432
base	13606	90671	84328	79256	103941
optgroup	13355	81027	77264	75849	101983
map	11560	71625	58206	54021	85112
article	9803	60360	70695	68550	69216
caption	8439	59366	55301	44789	79560
figure	7313	27054	34650	42734	38925
sub	6768	38054	27183	18992	29419
figcaption	6596	24062	31237	37629	34506
s	4905	32565	32671	39550	41208
acronym	4883	13833	17719	11935	18703
tfoot	4539	34842	28863	24570	34211
tt	4409	15966	19882	18799	35864
time	2952	34380	17614	11229	19127
canvas	2811	4054	9256	3748	28129
address	2568	13341	15244	10343	18145
menu	1818	5670	6454	4522	5994
details	1592	7630	11503	11496	15355
var	1399	11833	10365	7870	12197

Continued on next page

Table A.2 – continued from previous page

tagname	Jan 13 Total	Jan 14 Total	Jan 15 Total	Jan 16 Total	Jan 17 Total
dfn	1196	5867	5461	7470	8594
samp	1099	5327	7374	5813	5195
kbd	920	6193	5689	4612	8251
applet	563	2543	2281	2260	3142
hgroup	453	3842	3002	3216	4409
frame	449	4630	2950	3750	3335
q	290	1610	1737	1091	2025
noframes	284	3374	2251	2737	3061
frameset	252	2635	1597	2111	1978
source	122	1272	512	320	1920
video	81	312	215	207	659
bdo	38	83	16	88	81
summary	24	84	47	52	54
audio	14	84	112	159	110
dir	11	188	89	212	355
basefont	3	44	70	36	125
mark	1	#N/A	#N/A	23	1
rt	#N/A	634	332	261	1468
ruby	#N/A	607	216	144	1393
rp	#N/A	231	658	518	416
progress	#N/A	20	10	2	8
output	#N/A	#N/A	65	4	#N/A

Continued on next page

Table A.2 – continued from previous page

tagname	Jan 13 Total	Jan 14 Total	Jan 15 Total	Jan 16 Total	Jan 17 Total
datalist	#N/A	#N/A	#N/A	1	#N/A

REFERENCES

[1] K. Burton, N. Kasch, , and I. Soboroff, "The icwsm 2011 spinn3r dataset," in *In Proceedings of the Fifth Annual Conference on Weblogs and Social Media.* ICWSM, 2011. [Online]. Available: http://www.icwsm.org/data/

[2] W. Rao, L. Chen, P. Hui, and S. Tarkoma, "Move: A large scale keyword-based content filtering and dissemination system," in *Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on.* IEEE, 2012, pp. 445–454.

[3] TBD, "Towards a comprehensive picture of the great firewall's dns censorship," in *4th USENIX Workshop on Free and Open Communications on the Internet (FOCI 14).* San Diego, CA: USENIX Association, August 2014. [Online]. Available: https://www.usenix.org/conference/foci14/workshop-program/presentation/anonymous

[4] H. S. Hmood, Z. Li, H. K. Abdulwahid, and Y. Zhang, "Adaptive caching approach to prevent dns cache poisoning attack," *The Computer Journal*, vol. 58, no. 4, p. 973, 2015.

[5] P. Winter and S. Lindskog, "How china is blocking tor," 2012.

[6] "Iran; facebook users get 8-21 years in prison," 2014.

[7] R. McPherson, A. Houmansadr, and V. Shmatikov, "Covertcast: Using live streaming to evade internet censorship," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 3, pp. 212–225, 2016.

[8] X. Qiang. (2010) Translating the resistance discourse of chinese netizens. [Online]. Available: http://chinadigitaltimes.net/space/Grass-Mud_Horse_Lexicon_old_introduction

[9] B. Marczak, N. Weaver, J. Dalek, R. Ensafi, D. Fifield, S. McKune, A. Rey, J. Scott-Railton, R. Deibert, and V. Paxson, "An analysis of china's "great cannon"," in *5th USENIX Workshop on Free and Open Communications on the Internet (FOCI 15).* Washington, D.C.: USENIX Association, Aug. 2015. [Online]. Available: http://blogs.usenix.org/conference/foci15/workshop-program/presentation/marczak

[10] Z. Peng, "Online resistance to censorship among chinese fans of the big bang theory," *The Journal of Popular Culture*, vol. 49, no. 5, pp. 1023–1041, 2016.

[11] S. Li and N. Hopper, "Mailet: Instant social networking under censorship," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 2, pp. 175–192, 2015.

[12] J. Knockel, M. Crete-Nishihata, J. Q. Ng, A. Senft, and J. R. Crandall, "Every rose has its thorn: Censorship and surveillance on social video platforms in china," in *5th USENIX Workshop on Free and Open Communications on the Internet (FOCI 15).* Washington, D.C.: USENIX Association, Aug. 2015. [Online]. Available: https://www.usenix.org/conference/foci15/workshop-program/presentation/knockel

[13] R. R. Patil, *A method to evade keyword based censorship.* University of Texas at Arlington, 2014.

[14] A. Serjantov, "Anonymizing censorship resistant systems," in *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, ser. IPTPS '01. London, UK, UK: Springer-Verlag, 2002, pp. 111–120. [Online]. Available: http://dl.acm.org/citation.cfm?id=646334.687808

[15] C. Anderson, P. Winter, and Roya, “Global network interference detection over the ripe atlas network,” in *4th USENIX Workshop on Free and Open Communications on the Internet (FOCI 14).* San Diego, CA: USENIX Association, August 2014. [Online]. Available: https://www.usenix.org/conference/foci14/workshop-program/presentation/anderson

[16] L. L. Chin, U. Malone, and M. Khatami, “Iran bans facebook ahead of election: The iranian government has blocked access to the social networking site facebook as the country’s presidential election campaign gets under way,” 2009.

[17] F. Douglas and M. Caesar, “Ghostpost: Seamless restoration of censored social media posts,” in *6th USENIX Workshop on Free and Open Communications on the Internet (FOCI 16).* Austin, TX: USENIX Association, Aug. 2016. [Online]. Available: https://www.usenix.org/conference/foci16/workshop-program/presentation/douglas

[18] K. Boyd, “The effects of network security in non-censorship countries compared to countries with strong censorship,” p. 63, 2015, copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2016-06-06. [Online]. Available: https://login.ezproxy.uta.edu/login?url=http://search.proquest.com.ezproxy.uta.edu/docview/1752223394?accountid=7117

[19] P. Winter, S. Lindskog, K. universitet, A. fr datavetenskap, and k. o. I. Fakulteten fr ekonomi, “How the great firewall of china is blocking tor,” 2012.

[20] J. Wright, T. De Souza, and I. Brown, “Fine-grained censorship mapping: Information sources, legality and ethics.” in *FOCI*, 2011.

[21] 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.

[22] J. Biggs. (2015) Anti-censorship service greatfire is under attack. [Online]. Available: https://techcrunch.com/2015/03/19/anti-censorship-service-greatfire-is-under-attack/

[23] R. Singh, H. Koo, N. Miramirkhani, F. Mirhaj, P. Gill, and L. Akoglu, "The politics of routing: Investigating the relationship between as connectivity and internet freedom," in *6th USENIX Workshop on Free and Open Communications on the Internet (FOCI 16)*. Austin, TX: USENIX Association, Aug. 2016. [Online]. Available: https://www.usenix.org/conference/foci16/workshop-program/presentation/singh

[24] H. Kim and J. H. Huh, "Detecting dns-poisoning-based phishing attacks from their network performance characteristics," *Electronics Letters*, vol. 47, no. 11, p. 1, 2011.

[25] R. Fannin, "Why google is quitting china," *It's not censorship. The search giant just couldn't compete with Baidu.(15.01. 2010), Online under the URL: http://www. forbes. com/2010/01/15/baidu-china-search-intelligent-technologygoogle. html [As of: 11.05. 2013]*, 2010.

[26] J. Knockel, A. Senft, and R. Deibert, "Privacy and security issues in bat web browsers," in *6th USENIX Workshop on Free and Open Communications on the Internet (FOCI 16)*. Austin, TX: USENIX Association, Aug. 2016. [Online]. Available: https://www.usenix.org/conference/foci16/workshop-program/presentation/knockel

[27] R. Dingledine, N. Mathewson, P. Syverson, and N. R. L. W. DC, "Tor: The second-generation onion router," 2004.

[28] C. Connolly, P. Lincoln, I. Mason, and V. Yegneswaran, "Trist: Circumventing censorship with transcoding-resistant image steganography," in *Foci14*, 2014.

[29] D. Fifield and L. Tsai, "Censors' delay in blocking circumvention proxies," in *6th USENIX Workshop on Free and Open Communications on the Internet (FOCI 16).* Austin, TX: USENIX Association, Aug. 2016. [Online]. Available: https://www.usenix.org/conference/foci16/workshop-program/presentation/fifield

[30] F. Douglas, W. Pan, M. Caesar, *et al.*, "Salmon: Robust proxy distribution for censorship circumvention," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 4, pp. 4–20, 2016.

[31] Q.-U.-A. D. Akbar, M. Flores, and A. Kuzmanovic, "Dns-sly: Avoiding censorship through network complexity," in *6th USENIX Workshop on Free and Open Communications on the Internet (FOCI 16).* Austin, TX: USENIX Association, Aug. 2016. [Online]. Available: https://www.usenix.org/conference/foci16/workshop-program/presentation/akbar

[32] A. Houmansadr, G. Nguyen, M. Caesar, and N. Borisov, "Cirripede: circumvention infrastructure using router redirection with plausible deniability." ACM, 2011, pp. 187–200.

[33] E. Wustrow, S. Wolchok, I. Goldberg, and J. A. Halderman, "Telex: Anticensorship in the network infrastructure," in *Proceedings of the 20th USENIX Security Symposium*, Aug. 2011.

[34] M. Schuchard, J. Geddes, C. Thompson, and N. Hopper, "Routing around decoys," in *Routing around decoys.* ACM, 2012, pp. 85–96.

[35] M. Schuchard and N. Hopper, "E-embargoes: Discouraging the deployment of traffic manipulating boxes with economic incentives," *CoRR*, vol. abs/1606.08536, 2016. [Online]. Available: http://arxiv.org/abs/1606.08536

[36] I. Safaka, C. Fragouli, and K. Argyraki, "Matryoshka: Hiding secret communication in plain sight," in *6th USENIX Workshop on Free and Open Communications on the Internet (FOCI 16).* Austin, TX: USENIX Association,

Aug. 2016. [Online]. Available: https://www.usenix.org/conference/foci16/workshop-program/presentation/safaka

[37] S. Burnett, N. Feamster, and S. Vempala, "Chipping away at censorship firewalls with user-generated content."

[38] "Whiskey, weed, and wukan on the world wide web: On measuring censors' resources and motivations," in *Presented as part of the 2nd USENIX Workshop on Free and Open Communications on the Internet.* Berkeley, CA: USENIX, 2012. [Online]. Available: https://www.usenix.org/conference/foci12/workshop-program/presentation/Aase

[39] 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.

[40] G. Perng, M. K. Reiter, and C. Wang, *Censorship Resistance Revisited.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, vol. 3727, pp. 62–76.

[41] Z. Weinberg, J. Wang, V. Yegneswaran, L. Briesemeister, S. Cheung, F. Wang, and D. Boneh, "Stegotorus: a camouflage proxy for the tor anonymity system," in *StegoTorus: a camouflage proxy for the Tor anonymity system.* ACM, 2012, pp. 109–120.

[42] L. Invernizzi, C. Kruegel, and G. Vigna, "Message in a bottle: sailing past censorship," in *Message in a bottle: sailing past censorship.* ACM, 2013, pp. 39–48.

[43] G. Danezis and R. Anderson, "The economics of censorship resistance," in *In The Third Annual Workshop on Economics and Information Security (WEIS04*, 2004.

[44] T. Elahi, J. A. Doucette, H. Hosseini, S. J. Murdoch, and I. Goldberg, "A framework for the game-theoretic analysis of censorship resistance," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 4, pp. 83–101, 2016.

[45] J. J. Eberhardt, "Bayesian spam detection," *Scholarly Horizons: University of Minnesota, Morris Undergraduate Journal*, vol. 2, no. 1, p. 2, 2015.

[46] M. L. Mueller and H. Asghari, "Deep packet inspection and bandwidth management: battles over bittorrent in canada and the united states," *Telecommunications policy*, vol. 36, no. 6, pp. 462–475, 2012.

[47] J. J. Philip, "Prominent streaks discovery on blog articles," 2013.

[48] D. Rodriguez-Clark. (2013) Frequency analysis: Breaking the code. [Online]. Available: http://crypto.interactive-maths.com/frequency-analysis-breaking-the-code.html

[49] Stackoverflow.com. (2010) Regex match open tags except xhtml self-contained tags. [Online]. Available: http://stackoverflow.com/questions/1732348/regex-match-open-tags-except-xhtml-self-contained-tags

BIOGRAPHICAL STATEMENT

Kelly Scott French was born in Indiana, in 1968. He received his B.S. degree in Computer Information Science from Northeastern State University, Tahlequah Oklahoma in 1991 and his M.S. degree from The University of Texas at Arlington in 2016 in Computer Science. He is contemplating continuing with his PhD in the same major at University of Texas at Arlington.