CensMon: A Web Censorship Monitor

Andreas Sfakianakis, Elias Athanasopoulos, Sotiris Ioannidis
Institute of Computer Science
Foundation for Research and Technology, Hellas
{*sfakiana, elathan, sotiris*}*@ics.forth.gr*

Abstract

The Internet has traditionally been the most free medium for publishing and accessing information. It is also quickly becoming the dominant medium for quick and easy access to news. It is therefore not surprising that there are significant efforts to censor certain news articles or even entire web sites. For this reason, it is paramount to try to detect what is censored and by whom. In this paper we present the design and implemention of a web censorship monitor, called CensMon. CensMon is distributed in nature, operates automatically and does not rely on Internet users to report censored web sites, can differentiate access network failures from possible censorship, and uses multiple input streams to determine what kind of censored data to look for. Our evaluation shows that CensMon can successfully detect censored content and spot the filtering technique used by the censor.

1 Introduction

Censorship on the world wide web appears to be taking place more than ever before. The OpenNet Initiative (ONI) reports that there are almost 60 countries that either filter or are suspected of filtering web content [7]. Furthermore, in 2011, the Freedom House released a report that examined Internet-freedom in 37 countries around the world, and found that there is a diverse and growing threat to Internet-freedom [2]. Also, from 2000 onwards, many web censorship-related stories can be found by performing a simple web search in Slashdot's Your Rights Online section (YRO) [10].

Web users can learn about filtered content in countries from various articles on popular news websites. However, the information provided is usually very sparse and in most cases, limited to a few filtered URLs per country. Thus, there is a need for a service that provides more detailed information about up-to-date web censorship, such as ONI's Herdict Web [8].

Censorship is a phenomenon that changes over time. The main criteria used for censorship-checking by all these web services is whether a specific web site is accessible or not. ONI's Herdict Web [8], which is the major web censorship monitoring service, depends on user-generated feeds to determine the accessibility or inaccessibility of a web site. This fact alone may result in false positives, since often users cannot differentiate network failures from actual censorship. Therefore, there is need for a service that does *not* depend on user input, and runs transparently to track all the changes in the accessibility state of web pages.

In this paper, we present the design and implementation of CensMon, a system that offers users real-time information about filtered web content, without *actually* depending on their experience. Specifically our systems has three design characteristics: (i) it uses PlanetLab's [9] nodes, to create a worldwide web censorship monitor, (ii) it uses plug-in feed modules, that stream newly published, possibly sensitive, content to our system for censorship-checking, and (iii) we maintain historical data by continuously monitoring sites that have been censored. Our results, by using many different web sources, show that our system can exploit these sources to detect censored content and identify the filtering technique used.

2 Background and Related Work

Background about internet filtering mechanisms is provided in detail by Deibert et al. [14]. Wolfgarten [21] and Dornseif [15], apart from describing filtering techniques, provide information concerning filtering circumvention.

Zittrain and Edelman [23] have found a number of blocked websites in China associated with sensitive material. They used URLs from search results from web searches as input to test for blocking, making their probing more efficient and more targeted. In order to evaluate our system we used their methodology as one source of

our system's inputs.

Clayton et al. [12] give a detailed presentation of how web content blocking works. The main focus of their work is to reveal the mechanisms behind the Great Firewall of China. Moreover, they propose a naive but effective way to circumvent the Chinese firewall by just ignoring the injected TCP RST packets that the firewall generates. This work further motivates our effort and provide us with information about characteristics of the specific content blocking mechanism.

In 2007, Crandall et al. [13] proposed Conceptdoppler, an approach based on Latent Semantic Analysis (LSA) [17] to semi-automatically extract filtered keywords in China. In Conceptdoppler, words that were related to concepts that are deemed sensitive were extracted using LSA and then active measurements were conducted to evaluate their results. Moreover, Park and Crandall's latest work [19] is focusing on HTML response filtering and the discontinuation of this technique in China. Both these works influenced our methodology in order to spot URL keyword filtering and differentiate it from HTTP response keyword filtering and HTTP Header keyword filtering.

Mathrani et al. [18] tried to get a snapshot of censorship in 10 countries. Their design methodology of their probing mechanism influenced us in enabling us to detect the root cause of the filtering that was reported.

Xu et al. [22] use low-level network characteristics in order to detect the location of the filtering devices. To accomplish that, they use PlanetLab nodes for their probes, an approach that we follow as well.

The Open Net Initiative [7] is an organization that investigates and analyses Internet filtering. They provide country profiles that describe the social background and the reasons why censorship is employed, and release reports on different countries that censor the Internet. Moreover, they provide statistics about internet usage at each country, as well as a service called HerdictWeb [8] that informs web site visitors about what is censored in each one of these countries. Herdict Web allows one to see what is inaccessible, where it is inaccessible, and for how long. It uses crowd-sourcing to get information about censorship and present a real-time view of the experiences of users around the globe. Unfortunately, this service relies heavily on web users that can sometimes falsely report the inaccessibility of a site, raising a need for differentiation between censorship and network error. CensMon does not have access to as many nodes as Herdict, but it can work in a complementary fashion since it uses an automated mechanism.

Finally, there a few other existing applications that monitor Internet censorship events which are either focused on a specific country [5] or their agent network is still quite limited to have an accurate global view [6].

3 CensMon Architecture

CensMon is a system that conducts extensive accessibility tests, trying not only to detect the presence of filtering but also to spot the root cause of it, if possible. It consists of two basic building blocks: the central server and the network of sensing nodes. We refer to the sensing nodes as CensMon's agents. In this section, we describe the properties of CensMon's central server and demonstrate the design of our system by providing a test case.

3.1 Central Server

The central server is the core of the entire system. The web front-end runs there as well as all the scripts handling CensMon's input. Also it runs the database that stores all probing information and the filtering history.

3.2 Filtering Detection Procedure

We will now describe the methodology followed by our system. There are eight steps that take place in our system:

1. At the start, the central server receives as input a URL to test. It then forwards this URL to all alive agents in CensMon's network to be tested. To avoid a flood of messages to all agents, we have inserted a small time-out between messages that are sent to the agents.

2. Once an agent receives a URL from the server, its initial task is to make a DNS request for the domain of the URL so as to get the corresponding IP address or addresses. If no IP address is returned from the DNS server then the agent reports to the central server the probable cause of the DNS failure (e.g. connection refused, connection timed out or non-existing domain) to take further action.

3. If the agent successfully resolves the IP address of the domain in question, it tries to connect to that IP address at port 80, in an attempt to detect whether IP address blacklisting takes place. Upon successful connecting to the remote port, the agent continues to the next step of our protocol, otherwise it reports the connectivity problem to the central server.

4. Having determined there is no IP address blocking, the agent tries to find out if there is any kind of filtering at the application level (HTTP). The agent tries to detect URL keyword filtering. Inspired by Park's [19] approach, we have set up a seperate web server serving null (empty) content so as to avoid HTML response keyword filtering. The agent contacts our webserver requesting our

webserver's default URL concatenated with the URL requested from CensMon server. This is, if the requested URL is http://www.cnn.com, then our agent's HTTP request becomes http://www.ourdomain.com/www.cnn.com. This way the agent detects whether there is any kind of URL keyword filtering or the expected *404 Not Found* Status Code is returned. Again, if the agent detects URL keyword filtering it reports it to the central server.

5. Finally, the agent attempts to access the initial URL (using HTTP 1.1) and sends the received status code, HTML code and information gathered from the previous steps to the central server. In case of redirection the agent additionally reports the final URL and the final IP address visited.

6. Since all the agents have reported their findings for a specific probe, the central server starts its post-mortem analysis of the agents' reports. CensMon can detect filtering that uses DNS Name hijacking (probable redirection to a block-page) by using *whois* service and by correlating the resolved IP addresses (matching IP prefixes) returned by DNS servers to all our agents concerning a specific URL. Nevertheless, if a domain name has more than one associated IP addresses, then our data are not enough to determine DNS manilupation with precision.

7. Next, CensMon tries to identify censorship of partial content in a web page. Since the HTML code of a web page that was successfully accessed by one of our agents is stored at the server, CensMon analyses the HTML code of the web pages returned by all agents and are associated with the same URL. When the md5 hashings of the URLs' HTML code differ, CensMon uses Arc90's Readability functionality [1] in order to extract the content that is most likely to be the stuff a user wants to read (and what the censor wants to filter). Python port of arc90's readability traverses the DOM and uses a scoring function that rewards an element for containing text, punctuation, and class or id attributes typically associated with the main content of a site. This way CensMon deals with automatic changing/updated contents (such as news sites, e.g. nytimes.com) or contents that are localized (depending on where the user is coming from). CensMon uses fuzzy hashing [16] for comparing the URLs' *readable* HTML code and detecting partial filtering.

8. Lastly, when an inaccessibility event is reported, CensMon marks it as suspicious for filtering and begins to track the specific URL with the agent that reported the inaccessibility. This tracking is mandatory for CensMon to be able to differentiate between filtering, in cases where inaccessibility is repeatedly reported, network errors, if the URL finally becomes accessible, or change in censor policy.

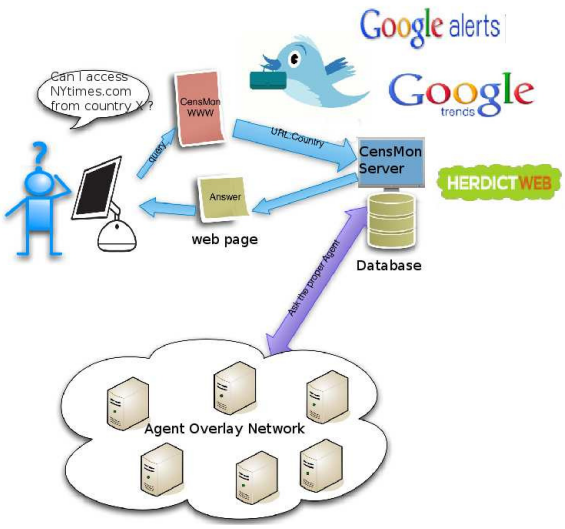

Figure 1: The CensMon Architecture.

The architecture of CensMon is illustrated in Figure 1. Figure 1 shows a user accessing CensMon's web front end, which is one of the possible systems inputs (in the upper right part of there figure are the inputs used during the evaluation period). The user can choose if they want to forward the URL in question to a specific alive agent or to all of CensMon's agents. Then the central server will forward the query and will try to detect possible filtering of the user request. Afterwards, server stores the results reported by the agents in the database. Finally, CensMon informs the user via web front-end, if a specific agent was chosen, or via email, if the entire agent nework was selected.

4 Evaluation

In this section we present the results of a preliminary evaluation of CensMon . First, we describe the experimental setup and then we present all the experimental results.

4.1 CensMon Agent Network

For our testbed we used nodes from PlanetLab. We run CensMon agents on PlanetLab nodes forming the CensMon network. More precisely, we have deployed 174 agents in 33 different countries (141 distinct ASes, 130

Country Code	#Agent Nodes	#ASes	Country Code	#Agent Nodes	#ASes	Country Code	#Agent Nodes	#ASes
AR	1	1	GR	2	1	NZ	2	2
BE	1	1	HK	2	2	PL	5	3
BR	5	3	HU	1	1	PT	2	1
CA	7	7	IE	1	1	RU	2	2
CH	5	1	IL	3	1	SE	2	2
CN	1	1	IT	2	1	SG	2	2
DE	15	4	JO	1	1	SI	1	1
ES	4	3	JP	10	8	TR	1	1
FI	1	1	KR	3	3	TW	5	4
FR	7	4	NL	3	2	US	72	72
GB	3	2	NO	1	1	UY	1	1

Table 1: Number of CensMon agents and number of ASes per country.

distinct cities) in PlanetLab. Table 1 shows the countries, the number of deployed agents in each one of them as well as the number of the corresponding ASes. In order to evaluate our system we used input from different sources. We now discuss how these sources provide CensMon with URLs to test.

User Input. CensMon has a front-end which enables users to insert URLs in the system. Users should specify the URL as well as the agent they want the system to forward the request to. After a successful request, users get the respective response and the HTML code of the requested URL. Another option is to insert a URL to be forwarded to all the CensMon agent network.

Google Alerts. Apart from user input, we use Google Alerts [4], a service provide by Google, for automatically inserting URLs of interest in our system. Google Alerts are email updates of the latest relevant Google results based on a topic of choice. The characteristics of an alert is the topic that we are interested, the frequency of receiving alerts (we choose to receive web alerts as they happen) and finally the type of the alert. Google Alerts offer five types of alerts for a specific topic: News alerts (related URLs from news sites), Blog alerts (URLs from Blogs), Real-time alerts (latest related Tweets), Discussion alerts (related threads from various fora) and Video alerts (newly published related videos).

Using Google Alerts as an input source we can check web content that may be censored and test how CensMon responds to this newly published content. We registered a Gmail account and added 4 topics to our alert services. These topics was *internet censorship*, *net neutrality*, *freedom of speech* and *human rights*. For each of these topics our email account receives alerts for all four type of alerts presented above. Using an IMAP client we fetch and insert all alerts to CensMon .

Internet Trends. In parallel, we want to test URLs that are associated with current trends discussed over the Internet. For this reason, we use the popular social network Twitter [11] and Google Hot Trends [3] for extracting periodically popular trends. Google Hot Trends [3] is a service provided by Google, where one can see a snapshot of people's interests. Nevertheless, since Twitter trends and Google Hot Trends do not include necessarily URLs, we extract characteristic keywords associated with these trends and then, by using the Google Search Engine, we feed CensMon with the top-10 URLs returned by Google for a given trend.

Herdict's Web Reported URLs. Since the OpenNet Initiative [7] is the best source of information for Internet censorship, we use as input the latest reported URLs by web users from the Herdict Web [8] site to test them with our infrastructure. We were periodically visiting Herdict Web site and automatically extracting the URLs that were reported by web users.

ONI's Categories for Internet Censorship. ONI [7] has released a list of categories in the global URL list for Internet censorship research. We chose ten of them in order to find related URLs and insert them to CensMon . The ten categories that we selected are: news outlets, freedom of speech, entertainment, government, terrorism, porn, gambling, religion, net neutrality and human rights. We then proceed and search through Google Search Engine to find the top-100 results for each category. The resulting 1000 URLs of the above categories were inserted to CensMon so as to be tested through the agent network.

4.2 Experimental Results

All evaluation measurements were conducted during a 14-day period in April 2011. At this period CensMon tested 4950 unique URLs from 2500 domains. Moreover, CensMon detected 951 unique URLs from 193 domains as filtered. Table 2 depicts the number of unique domains found as censored by CensMon during the evaluation period.

AR	0	GR	0	NZ	0
BE	0	HK	2	PL	0
BR	0	HU	1	PT	0
CA	0	IE	1	RU	0
CH	0	IL	0	SE	0
CN	176	IT	0	SG	0
DE	1	JO	5	SI	0
ES	0	JP	1	TR	6
FI	0	KR	0	TW	0
FR	0	NL	0	US	0
GB	0	NO	0	UY	0

Table 2: Censored domains found per country by CensMon.

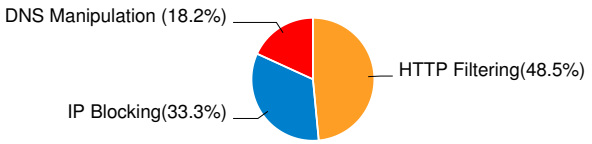

Figure 2: Percentage breakup of the layer at which the filtering was detected during CensMon 's evaluation period.

In Figure 2 we can see the percentage of the filtering techniques concerning the censored URLs found during the evaluation period. Furthermore, CensMon reported that for the domains that was detected filtering at HTTP level in China, 71% of these domains were censored due to URL keyword filtering. Finally, throughout the evaluation period CensMon did not detect any partial filtering of a web page.

Figure 3 shows the distribution of the unique domains that CensMon has detected as censored for all the agent nodes. As we can see, about 86% of the agent nodes have not reported any filtering event to be categorized as censored by CensMon . Moreover, about 10% of the agent nodes have found 1 to 6 domains as censored. The Chinese agent node was by far the one that reported filtering in 176 domains marked as censored.

CensMon can detect whether an inaccessibility was reported due to temporary failure or filtering after a number of tracking attempts. Whenever our system gets information by one of the agents that a specific URL is inaccessible, it tracks it in order to spot the differentiation between filtering and a possible network error events. As Figure 4 depicts, 21% of all the URLs that CensMon started to track were accessible after the first tracking attempt and during *all* the rest of the evaluation period, concluding that the initial inaccessibility had been caused due to network failure. Moreover, the decrease of the percentage after the first tracking attempt can be explained due to the fact that the very first attempt is done by all

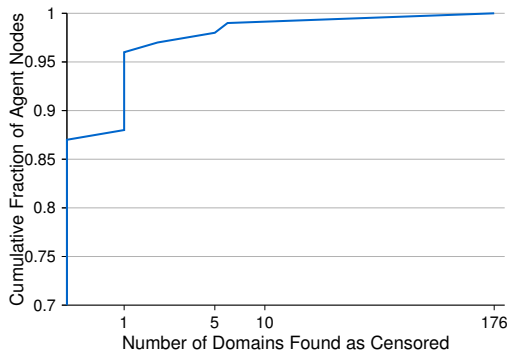

Figure 3: Cumulative distribution of the unique censored domains found during evaluation period.

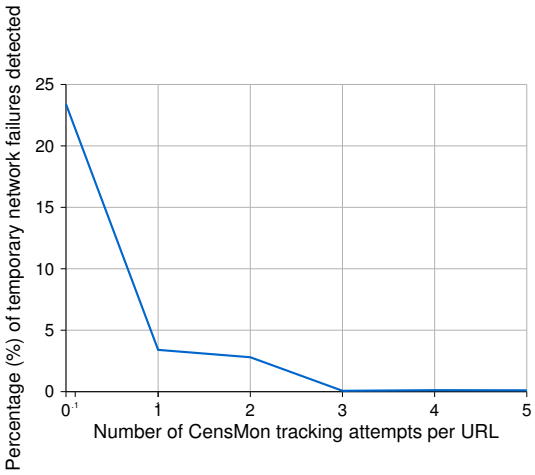

Figure 4: Percentage of temporary network failures detected after a varying number of CensMon tracking attempts.

agent nodes trying to reach the same web server, while tracking attempts are more lightweight. Therefore, after 3 tracking times we can be confident that what CensMon tracks is rather a filtering event than an early and temporary network failure. However, we have tested manually these 193 domains that were reported as censored in order to check for false positives and we have found that 3 of them were falsely marked.

Finally, 123 distinct URLs that were inserted as input to CensMon through ONI's Herdict Web site. CensMon started to track 245 cases that an inaccessibility was reported coming from 75 agent nodes in 20 countries. At last, 192 of the 245 cases of reported inaccessibility were at least one time accessible from our system while the rest 53 cases were reported as inaccessible during all our evaluation period.

5 Discussion

One limitation of our work for now is the fact that CensMon can only monitor the extent of censorship of a country within which we are able to access a PlanetLab node. However, we can overcome this issue by using web proxies worldwide as agents conducting simple accessibility tests or by developing a software client or even a Firefox add-on that will act as a CensMon agent.

Another direction that we plan to explore further is monitoring specific news sites via RSS feeds, and measure Internet censorship concerning realtime news events. Moreover, we can use our agent network to measure filtering of non-HTTP traffic and ports (e.g. P2P, SMTP, VPN etc.) or execute network level probes in order to test network infrastructure in terms of censorship.

As Rogers proposed in [20], we can make use of CensMon's infrastructure to find known blocked content on unblocked sites within a country.

Finally, by conducting long-term measurements and since in CensMon each URL is forwarded to all its agents, we could detect common domains censored by different countries. As a result, we can extract information about common worldwide web filtering trends among countries.

6 Conclusion

In this paper we presented CensMon, an Internet censorship monitoring infrastructure. The increase of global web censorship motivated us to design and build a system that can detect filtering characteristics and also be capable of differentiating between censorship and network failures. We implemented our design and evaluated it on the PlanetLab testbed using information streams automatically extracted from a plethora of sources. Based on our experience with using CensMon, as well as on the experimental results presented in this paper, we believe that CensMon can be a valuable resource for Internet censorship detection, and can provide useful information for both researchers and regular web users.

Acknowledgements

This work was supported in part by the Marie Curie Actions – Reintegration Grants project PASS. Elias Athanasopoulos is funded by the Microsoft Research PhD Scholarship project, which is provided by Microsoft Research Cambridge.

References

[1] Arc90's Readability. `http://www.readability.com`.

[2] Freedom House. Freedom on the Net 2011. `http://www.freedomhouse.org/template.cfm?page=664`.

[3] Google Hot Trends. `https://www.google.com/trends/hottrends`.

[4] Google Web Alerts. `http://www.google.com/alerts`.

[5] GreatFirewall.biz. `http://www.greatfirewall.biz`.

[6] Net Neutrality Monitor. `http://www.neumon.org`.

[7] Open Net Initiative. `http://www.opennet.net`.

[8] OpenNet Initiative's Herdict Web. `http://www.herdict.org/web/`.

[9] PlanetLab. `http://www.planet-lab.org`.

[10] Slashdot. Your Rights Online. `http://yro.slashdot.org`.

[11] Twitter. `http://twitter.com/`.

[12] R. Clayton, S. Murdoch, and R. Watson. Ignoring the great firewall of china. In *Privacy Enhancing Technologies*, pages 20–35. Springer, 2006.

[13] J.R. Crandall, D. Zinn, M. Byrd, E. Barr, and R. East. Conceptdoppler: A weather tracker for internet censorship. In *14th ACM Conference on Computer and Communications Security*, 2007.

[14] R. Deibert. *Access denied: the practice and policy of global Internet filtering*. The MIT Press, 2008.

[15] M. Dornseif. Government mandated blocking of foreign Web content. *Arxiv preprint cs/0404005*, 2004.

[16] J. Kornblum. Identifying almost identical files using context triggered piecewise hashing. *digital investigation*, 3:91–97, 2006.

[17] T.K. Landauer, P.W. Foltz, and D. Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2):259–284, 1998.

[18] A. Mathrani and M. Alipour. Website Blocking Across Ten Countries: A Snapshot. 2010.

[19] J.C. Park and J.R. Crandall. Empirical Study of a National-Scale Distributed Intrusion Detection System: Backbone-Level Filtering of HTML Responses in China. In *2010 International Conference on Distributed Computing Systems*, pages 315–326. IEEE, 2010.

[20] R. Rogers. A New Media Approach to the Study of State Internet Censorship. `http://www.govcom.org`, 2009.

[21] S. Wolfgarten. Investigating large-scale Internet content filtering. *M. Sc. in Security and Forensic Computing*, 2006, 2005.

[22] X. Xu, Z. Mao, and J. Halderman. Internet Censorship in China: Where Does the Filtering Occur? In *Passive and Active Measurement*, pages 133–142. Springer, 2011.

[23] J. Zittrain and B. Edelman. Internet filtering in China. *Internet Computing, IEEE*, 7(2):70–77, 2003.