Blocking-Resistant Protocol Classification Using Bayesian Model Selection

Brandon Wiley

School of Information, University of Texas at Austin
1616 Guadalupe #5.202
Austin, TX 78701-1213
brandon@ischool.utexas.edu

Abstract. Internet censorship techniques have become increasingly sophisticated by utilizing Deep Packet Inspection for the classification and filtering of connections by protocol. The response from circumvention tools has been to implement blocking-resistant protocols that resemble existing protocols such as SSL or new protocols such as that used by obfsproxy. So far this approach has created periods of time where censorship is circumvented between when the new protocol is deployed and new filtering rules are written to distinguish encoded circumvention traffic from non-circumvention traffic. This paper describes a method for dynamic classification of protocols based on Bayesian models created from sampled traffic. This approach allows for the automated testing of blocking-resistant protocol designs against simulated passive attackers.

Keywords: censorship resistance, blocking resistance

1 Introduction

The evolution of Internet censorship has been a cycle of increasingly sophisticated filtering techniques inspiring new circumvention techniques. Shallow Packet Inspection filtered based on the IP addresses in packet headers, leading to the development of anonymizing proxy networks that hide the true destination IP address by first routing through proxies. Filtering technology is now employing Deep Packet Inspection (DPI) techniques that can filter out specific Internet protocols. [13] This has resulted in censorship-resistant services such as anonymizing proxies being specifically targeted to

be blocked or throttled. In order to provide censorship resistance, blocking resistance is now necessary to defeat DPI filtering.

Currently, new blocking-resistant protocols are developed in response to filtering rules that are created to detect the blocking-resistant protocols. Once the encoded traffic can be identified from other traffic, it is easily blocked by manually coded firewall filtering rules. However, before it can be blocked, it must be identified by protocol identification experts that have examined the network traffic.

An alternative approach to manual development of firewall filtering rules is to use an automatic classifier. Given labeled samples of traffic generated by different protocols, a computational model can be constructed which can generate predictions for traffic samples with characteristics matching those of each protocol. Unlabelled traffic data can then be compared against each model and the most probable model can be selected. This allows for probabilistic classification of traffic without the need for manual intervention. The benefit of this approach is that it allows multiple obfuscating protocols to be tested for effectiveness before being deployed in the field. While this approach cannot guarantee that a obfuscating protocol is undetectable, it offers a first pass for evaluating obfuscating protocols by ensuring that they are not easily distinguishable by examining statistical distributions of key properties such as length, timing, and entropy. In order to evaluate the efficacy of the obfuscating protocols against blocking, a framework for evaluating blocking-resistant transports was created. The framework, simply called “blocking-test”, provides a fully automated testing solution from traffic generation to reporting.

2 Related Work

While a network of proxy nodes can provide protection against destination IP blacklists, they are still vulnerable to various forms of DPI protocol fingerprinting. This problem is dealt with by Kopsell, who proposes a method to extend existing anonymous publishing systems to bypass blocking, a property referred to as "blocking resistance" [5]. Kopsel’s threat model assumes that the attacker has control of only part of the Internet (the censored zone), that some small amount of unblockable inbound information can enter the censored zone (perhaps out of band). The nodes in Kopsell’s system are volunteer anonymizing proxies that clients communicate with over a steganographic protocol in order to obtain access to a censorship-resistant publishing system. Clients obtain an invitation to the network, including the IP addresses of some proxy nodes, through a low-bandwidth, unblockable channel into the censored zone.

Kopsell's proposal used SSL as the communication channel. Unfortunately, while SSL does not offer blocking resistance when SSL traffic is specifically targetted. Tor has suffered blocking by two attacks because of its use of SSL. One targeted unique characteristics of Tor's SSL handshake specifically and the other was a throttling of all SSL traffic [2][9]. While Tor has subsequently increased the steganographic strength of its SSL handshake by making it resembled Apache's SSL handshake, this does not prevent against an attack that blocks or throttles all SSL traffic.

2.1 Obfuscated Protocols

An obfuscated protocol, in contrast to a secure protocol, provides protection from the attacker only so long as the attacker does not know the details of the encoding. For instance, BitTorrent clients have implemented three obfuscating protocols in order to prevent filtering and throttling of the BitTorrent protocol, the most common of which in current usage is Message Stream Encryption (MSE) [7]. Analysis of packet sizes and the direction of packet flow have been shown to identify connections obfuscated with MSE with 96% accuracy, primarily through analysis of the statistical properties of the key exchange [4].

Obfuscated TCP (ObsTCP) has gone through several versions, the last of which used DNS records to transmit the encryption keys [8]. This required the attacker to correlate separate communication streams, extracting the keys from the DNS packets and then applying them to the TCP packets. However, an analogous attack has already been demonstrated in the blocking of BitTorrent traffic through monitoring of the tracker protocol traffic to obtain the ports of the BitTorrent protocol connections [10][3]. A similar proposal called tcpcrypt is also easily defeated by looking for static strings in the handshake [1].

The most sophisticated option is obfuscated-openssh, which replaces the SSH handshake portion of an SSH protocol connection with a minimal blocking-resistant encrypted protocol [6]. This handshake is encrypted with a key that is generated by iterated hashes of a seed that is added to the beginning of the encrypted part of the handshake. The iteration number is chosen to be high enough that key generation is slow, so the blocking resistance of this technique relies on key generation being too expensive to scale to all connections simultaneously. However, modern filters are capable of statistically sampling packets and processing them offline [11].

Dust is an Internet protocol designed to provide blocking resistance against DPI techniques. [12] Dust uses a novel out-of-band handshake to establish a secure, blocking-resistant channel for communication over a filtered channel. Once a secure channel has been established, Dust packets are indistinguishable from random packets and so cannot be filtered by normal

techniques. For attackers that filter random packets an optional unrandomization step is used to give the packets arbitrary statistical properties.

3 Methodology

The first step in building the statistical models was to generate sample traffic. Since the majority of Tor traffic is presumed to be web traffic, traffic was generated using a Firefox web browser driven by a Selenium script, allowing for realistic web traffic including traffic from embedded images, CSS, and dynamic elements driven by Javascript such as AJAX connections. Firefox was selected because it can have its proxy configuration programmatically altered by the Selenium script. It could therefore be configured to generate either normal HTTP/HTTPS traffic or to route the web traffic through a proxy.

Three different configurations were used to generate traffic: no proxy, using Tor with obfsproxy as a proxy, and using Dust as a proxy. The trace from the traffic generated by Firefox fetching the web pages using these configurations was captured, divided into individual streams, tagged based on protocol by filtering to include only the packet sent to ports of interest, either SSL or obfsproxy or Dust, and put into traffic capture files. These files were then processed to extract the characteristics used by the models. These characteristics were packet length, relative packet timing (the interval between packets), and entropy. For length and timing, the values for all packets were extracted. For entropy only the first packet was used as the protocols that were examined are encrypted after their handshakes and so have maximum entropy after the first few packets. The first packet therefore serves as a better indicator than would entropy analysis of the entire packet stream.

The processed packet streams were then used to build Bayesian predictive models. The length detector used a multinomial likelihood with a frequency count for each observed packet length from 0 to 1500 and an uninformative Dirichlet prior. The timing model used a Poisson likelihood to generate each interval between packets in milliseconds, with an uninformative gamma prior. The entropy model used a Gaussian likelihood to represent a single entropy number for the first packet, with a Gaussian prior for the mean and a uniform prior from 0 to 100 for the standard deviation.

The Bayesian models, once trained on the observed data, were used to generate predictions. For each model 1000 predictions were generated. For the length model the predictions were 1400 counts, one for each of the packet lengths 0 to 1500, which was the maximum observed packet length (and a common MTU). For the timing model the predictions were 1000 packet

timing intervals. For the entropy the predictions were a single number indicating the entropy measurement of the first packet.

Once the predictions were generated, new traffic was captured and processed. The empirical observations from the new traffic were compared against the predictions from the models. The difference between the empirical and predicted traffic was squared to generate a match score. Scores were calculated over all 1000 predictions for each model. The model with the highest cumulative score was selected as the most probable match for the new traffic being analyzed. Model selection was done independently for each packet characteristic so that both the detectability of the protocol and the effectiveness of each detector could be measured. Finally, scores were generated showing the accuracy of each detector, the effectiveness of each encoder including false positives and false negatives, and the mutual distinguishability of each pair of protocols.

5. Results

Three protocols were tested: Dust, SSL, and obfsproxy, an implementation of obfuscated-openssh which acts as a pluggable transport for the Tor protocol. Three detectors were used: length, timing, and entropy. Overall across all three protocols, the length detector was 16% accurate, the timing detector was 89% accurate, and the entropy detector was 94% accurate at correctly classifying protocols. This was particularly interesting as the length and timing detectors used the entire packet streams, whereas the entropy detector achieved superior accuracy using only the first packet.

In terms of the different protocols, SSL was correctly identified 25% of the time, 70% of the time other protocols were misidentified as being SSL, and 5% of the time SSL was misidentified as another protocol. obfsproxy was correctly identified 55% of the time, 10% of the time other protocols were misidentified as being obfsproxy, and 35% of the time obfsproxy was misidentified as another protocol. Dust was correctly identified 48% of the time, 4% of the time other protocols were misidentified as being Dust, and 48% of the time Dust was misidentified as another protocol. In terms of protocol distinguishability across dectectors, obfsproxy was distinguishable from SSL 96% of the time, obfsproxy was distinguishable from Dust 98% of the time, and Dust was distinguishable from SSL 56% of the time.

In terms of overall distinguishability, the results of the evaluation show that Dust is less distinguishable from SSL than obfsproxy. In fact, Dust was never correctly identified the length and timing detectors. Only the entropy detector was able to identify Dust traffic. Overall, Dust was correctly identified 48% of the time and obfsproxy 55% of the time, showing an 7% difference in distinguishability across all detectors.

6. Conclusion

A Bayesian modeling approach was implemented for the automatic classification of network traffic into protocols using supervised learning from sample traffic. This method proved to be an effective means of distinguishing SSL, obfsproxy, and Dust traffic. It allowed for the effectiveness of different detectors to be compared, as well as the effectiveness of different encodings at defeating these detectors.

Evaluation of the obfuscating protocols through the construction of Bayesian models of the different packet characteristics used by DPI filters has shown that different encodings provide different levels of effectiveness in resisting against filtering. A protocol such as Dust that provides obfuscation of multiple packet characteristics was shown to be more effective than a protocol such as obfsproxy that provides only encryption of the packet contents. As packet length and timing were shown to be effective methods of classifying packets by protocol, future blocking-resistant protocols should incorporate measures to hide these packet characteristics. Additionally, entropy measurement, a type of filtering not currently in widespread use, was shown to be effective and inexpensive to implement when only the first packet of each conversation is used. Future obfuscating protocols, need to incorporate defenses against this type of filtering as well.

Those wishing to examine or use the blocking-test framework for academic or practical purposes can find the source code for its implementation at https://gitweb.torproject.org/user/blanu/blocking-test.git.

7. Acknowledgments

This work was made possible by support of Google Summer of Code, the Tor project, the Electronic Frontier Foundation, and Luis Francisco-Revilla.

References

1. Bittau, A., Hamburg, M., Handley, M., Mazieres, D., and Boneh, D. The case for ubiquitous transport-level encryption. 19th USENIX Security Symposium., (2008).
2. Dingledine, R. Tor and circumvention: Lessons learned. The 26th Chaos Communication Congress, (2009).
3. Harrison, D. BEP 008: Tracker Peer Obfuscation. Retrieved from: http://www.bittorrent.org/beps/bep_0008.html.

4. Hjelmvik, E and John, W. Breaking and Improving Protocol Obfuscation. Department of Computer Science and Engineering, Chalmers University of Technology, Technical Report No. 2010-05, ISSN 1652- 926X. (2010)
5. Kopsell, S., Hilling, U.: How to Achieve Blocking Resistance for Existing Systems Enabling Anonymous Web Surfing. In: Proceedings of the Workshop on Privacy in the Electronic Society. pp. 103-115. ACM Press, New York (2004)
6. Leidl, B. Obfuscated-OpenSSH README. Retrieved from: https://github.com/brl/obfuscated-openssh/blob/master/README.obfuscation. (2010)
7. Message Stream Encryption. http://wiki.vuze.com/w/Message_Stream_Encryption (2006)
8. Obfuscated TCP. Wikipedia. Retrieved from: http://en.wikipedia.org/wiki/Obfuscated_TCP. (2010)
9. Sennhauser, M.: The State of Iranian Communication. http://diode.mbrez.com/docs/SoIN.pdf (2009)
10. Topolsky, R. Comments of Robert M. Topolsky In the Matter of Petition of Free Press et al. for Declaratory Ruling that Degrading an Internet Application Violates the FCC's Internet Policy Statement and Does Not Meet an Exception for "Reasonable Network Management". Federal Communications Commission WC Docket No. 07-52, 08-7. (2008)
11. Using NetFlow Filtering or Sampling to Select the Network Traffic to Track. Retrieved from: http://www.cisco.com/en/US/docs/ios/netflow/configuration/guide/nflow_filt_samp_traff.html#wp1064305. (2006)
12. Wiley, B. Dust: A Blocking-Resistant Internet Transport Protocol. http://blanu.net/Dust.pdf (2011)
13. Winter, P. and Lindskog, S.: How China Is Blocking Tor. Karlstad University. (2012)