Traffic Flow Analysis of Tor Pluggable Transports

Khalid Shahbar A. Nur Zincir-Heywood
Faculty of Computer Science
Dalhousie University
Halifax, Canada
{Shahbar, Zincir}@ cs.dal.ca

***Abstract*— Tor provides the users the ability to use the Internet anonymously. On the Tor network, the users connect to three relays run by volunteers. The addresses of these relays are publicly available. Some organizations prevent access to Tor by blocking the addresses of these relays. To mitigate this, Tor has introduced the concept of bridges and pluggable transports. Bridges are relays that do not have publicly available addresses so that they can evade the blocking. Pluggable transports are used to obfuscate the connection to these bridges. In this paper, we investigate the robustness of these pluggable transports in evading the flow based traffic analysis and blocking systems.**

Keywords—Tor; Pluggable Transports; Traffic metadata; Traffic flow analysis

I. INTRODUCTION

The Tor network provides its users the ability to anonymously use the the Internet. It hides the users' identities from the websites they browse. It also hides the websites visited by the Tor users from an observing entity such as censorship or other attacks against the Tor network. This anonymity on the Tor network drove the censorships to search for different ways to block or to detect the Tor traffic. This starts by blocking the IP addresses of Tor relays run by volunteered users. Therefore, the Tor network provides its users with the ability to bypass such blocking by connecting to special relays, called bridges. Bridges do not have publicly available addresses [1]. However, this does not prevent censorships discovering these bridges and blocking them [2]. To this end, traffic classification (detection) is used to discover the Tor users connecting to a bridge. In return, the Tor network designed and developed the concept of Pluggable transports [3] to form the connection to the bridge and make it look like something different from Tor traffic. Therefore, using pluggable transports offers the Tor users access to the bridges, when the bridges are blocked by the censorship using their IP addresses.

The pluggable transports systems work different than other tools to provide access to the Tor network. Most of the other tools concentrate on hiding the content of the packets in a way that makes it hard to use deep packet inspection (DPI) to detect the connection to the bridges. But DPI is not the only method used to detect Tor traffic. Active probing and flow analysis are some of the other popular methods used to detect Tor traffic. Thus, in this research, we investigate the robustness of the pluggable transports in evading such methods.

The rest of this paper is organized as follows. Related work is reviewed in section II. The background of pluggable transports is summarized in section III. Section IV details the experiments performed whereas the results and the discussions are presented in section V and VI, respectively. Finally, the conclusions are drawn and the future work is discussed in section VII.

II. RELATED WORK

Classifying different types of encrypted traffic is a subject of many researches. Hjelmvik and John [4] used statistical analysis to do protocol classification. In their experiment, they classified the following protocols using SPID algorithm: BitTorrent, eDonkey, HTTP, SSL, and SSH. The results showed a recall rate of approximately 77%.

La Mantia et al. [5] classified different applications by using stochastic packet inspection. They proposed to use multiple flows for the same application and the same two communication end points. They aim to extract the right set of features for the purpose of application classification. In their experiments, they included HTTP, FTP, IMAP, POP, Skype, SMTP, SSH, and other protocols. Their results showed an average of 98% of true positive rate.

Barker et al. [6] run a simulation to distinguish Tor traffic from HTTPS traffic. By using three machine learning algorithms (Random Forest, J48, and Adaboost), they could distinguish between the three classes (HTTPS, HTTP over Tor, HTTPS over Tor) with approximately an average of 96% true positive rate in their simulation.

In our previous paper [18], we implemented two different techniques to identify the type of application within the encrypted Tor traffic: Circuit level classification and Flow level classification. We compared different ML algorithms and flow exporter tools to achieve the best possible results. The evaluations showed that the C4.5 decision tree classifier and the Tranalyzer flow exporter performed the best. Based on those results, in this paper, we used Tranalyzer as the flow exporter and the C4.5 decision tree as the classifier.

In doing so, we aim to evaluate the implemented Tor pluggable transports on a real network environment and explore the robustness of these pluggable transports in evading the flow based traffic analysis and censorship system.

III. Pluggable Transports

Pluggable transports have become a necessary requirement for some of the Tor network users who cannot have access to the Tor bridges [8] [9]. Tor provides the framework (Obfsproxy [13]) to developers to integrate their obfuscating tools with Tor, or to write a completely new obfuscating proxy for Tor.

A. *Flashproxy*

FlashProxy [10] is a proxy tool devolved using JavaScript. It allows the Tor user to access the Tor relay using continuously changing IP addresses. These addresses are used to present the visitors to the FlashProxy supported websites. A Flashproxy JavaScript code is included in these websites in order to provide the Flashproxy services to the Tor users.

B. *Scramblesuit*

Scramblesuit [11] works within the Obfsproxy that Tor provides for pluggable transport plugin. This tool works with the transport layer to obfuscate the application used. Authors claim that Scramblesuit resist the active probing and the flow signature used by the censorships to block the services from the users.

C. *Format-Transforming Encryption (FTE)*

This tool changes the encrypted traffic to look like another protocol such as HTTP. FTE [12] mainly works to evade the DPI method from identifying the protocol. It takes any Regex that the user wants the traffic to look like. Then lets the ciphertext to follow the shape (traffic form) of this regex. This way, the DPI identifies the ciphertext as the regex that the user wants it to be (such as HTTP).

D. *Meek*

Meek [14] uses popular websites (Google – Amazon – Azure) to redirect the user request to the Tor bridge. The Tor client sends a HTTPS request to one of these sites. The header of the HTTPS contains the required connection to the bridge running the meek server. The main idea behind this method is to use domains (such as Google, Amazon etc.) that are not (most likely) blocked. The user can also configure meek to use other websites other than the three default ones used by Tor. This requires that a Content Delivery Network (CDN) is also configured in the new domains that are used.

E. *Obfs3*

Obfs3 [15] is an obfuscator for the TCP protocol layer. It is used by Tor to prevent content analysis to discover Tor. Obfs3 uses modified Diffie Hellman key exchange. The use of Diffie Hellman is to enhance the security level for the key exchange and prevent compromising of the key during the exchange process. Obfs3 does not change the data length. It mainly focuses on hiding Tor characteristics to make it hard to detect by content search used by the censorships.

IV. Experiments And Evaluation

In our experiments, we configured four virtual machines and one Ubuntu Desktop 12.04. All the machines configured to use one pluggable transport at a time to connect to the Tor network. The traffic data are collected from these five machines. Once the machines connected to the Tor network, an automated script starts to browse different websites then closes the connection after the browsing (or watching the videos etc.) activities are completed. This process repeats until we collect sufficient amount of data. We used Tranalyzer [16] to extract the flows and Weka [17] for classification. The following details the experiments for each pluggable transport and the amount of data collected:

A. *Obfs3Traffic*

The data for the Obfs3 bridge connection have been collected from connections to two bridges. The first bridge was configured by using the recommended bridge setting in the Tor browser (Obfs3). The port used in this bridge was port 80 (one of the well-known ports assigned for HTTP). Even though the flow characteristics do not depend on the port number to identify the type of protocol used in the connection, HTTP traffic is included in our background traffic to compare the ability of the classifier to distinguish between two different applications while both use the same port number.

The second bridge was configured by running a node as a bridge and could accept Obfs3 connections. Then four virtual machines running Ubuntu Desktop 12.04 were configured to use our Obfs3 bridge to connect to the Tor network. The port number used was a dynamic port number. The total amount of Obfs3 traffic captured in our experiments is ~20 GB with 16953 flows.

B. *FTE Traffic*

The FTE data were collected from five machines with Ubuntu Desktop 12.04 as the operating system. Four of them were running virtual machines. The data were collected via connecting to five different FTE servers. The total amount of FTE traffic collected is ~23 GB. The number of collected flows is 106549.

C. *Scramblesuit Traffic*

In addition to active probing DPI resistance, Scramblesuit is designed to resist flow analysis by generating different flows for every Scramblesuit server. For this reason, we tried to collect different flows from multiple Scramblesuit servers. By using the bridge database, we collected our Scramblesuit data from connecting to 22 different Scramblesuit servers. The importance of having different servers is to have a variety of behaviors based on the design of Scramblesuit that changes the server flow for every server. The total number of flows collected from these 22 servers is 10649. The total amount of Scramblesuit traffic collected is ~22 GB.

TABLE I. EVALUATION RESULTS USING 10-FOLD CROSS VALIDATION.

	Class	TP Rate %	FP Rate %	Pre-cision %	Recall %	F-Measure %
Background Traffic	***HTTP***	99	0.1	99	99	99
	HTTPS	94	0	95	94	95
	SSH	99	0	99	99	99
	BT	94	2.5	84	94	89
	BTecr	89	0.9	96	89	92
Pluggable Transports Traffic	***FTE***	99	0	99	99	99
	Scramble suit	98	0.1	92	98	95
	Meek	99	0	99	99	99
	Flash proxy	99	0.1	99	99	99
	Obfs3	99	0	99	99	99
Overall Correctly Classified Instances	***97%***					

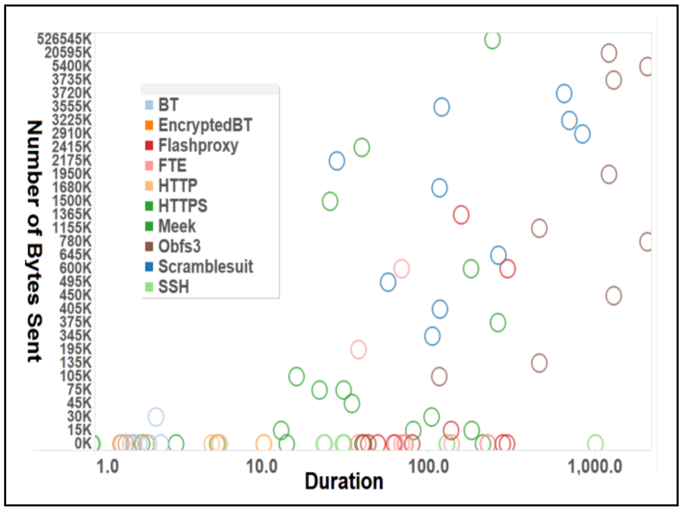

Fig. 1 Duration vs Transferred data

D. *Meek Traffic*

Meek makes connections with popular websites that provide services used by a wide range of users. These services include Google, Amazon, and Azure. For example, when Google is used as the front domain for Meek, then multiple addresses appear with this setting all belong to Google. In our experiments, the total number of flows is 43152. The data size is ~22 GB.

E. *Flashproxy Traffic*

In Tor, usually user starts the connection to the bridge. However, in Flashproxy, it is the other way around; the Tor user will receive connections from the visitors of the Flashproxy supported websites. This requires that the user has the ability or the access to do port forwarding if he/she is behind a NAT or has an open port configured to listen for incoming connections. The number of connections is high compared to the other pluggable transports. In our experiments, the total number of Flashproxy flows is 172331. The data size is ~ 11 GB.

F. *Other Traffic*

Pluggable transports are used by Tor to obfuscate Tor traffic in different flavors. To study the efficiency of these tools, pluggable transports traffic should be compared to the flavor of traffic they are trying to mimic and with different types of encrypted traffic. Thus, we added five different types of traffic as the background non-Tor (normal) traffic as follows: ~26 GB of peer-to-peer BitTorrent traffic, ~24 GB of encrypted BitTorrent traffic also collected, ~29 GB of SSH traffic, ~1 GB of HTTPS (SSL) traffic, and ~0.5 GB of HTTP traffic.

V. RESULTS

We performed evaluations using the 10-fold cross validation technique on the datasets. The detailed results of this approach are shown in Table I. The amount of data transferred in a specific duration depends on the applications that are generating the traffic. For example, HTTP traffic tends to have low duration and low traffic volume. BitTorrent has higher volume than HTTP because of the file sharing and has low to medium duration. On other hand, Obfs3 has high duration and the high traffic volume (data transfer). This is because when the user connects to Obfs3 bridge, the connection stays active as long as the user is using Tor. Fig. 1 shows a sample of 100 instances; 10 instances for every traffic type from our data set.

VI. DISCUSSION

In our experiments, we observed that the pluggable transports could have different flow behaviors than the other types of traffic as detailed in the following subsections.

A. *Number and repetition of connections*

It is noted that the number of connections made by a Tor user, when configuring a Tor browser to use the pluggable transport, reflects the type of pluggable transport. Scramblesuit and Obfs3 users make connections to one bridge during the duration of the user connection to the Tor network. In contrast, Flashproxy user receives multiple connections over a short period of time from multiple IP addresses. This is based on the mechanism that Flashproxy uses and makes using Flashproxy hard to block as these connections live for a short period and continuously come from different IP addresses. On the other hand, a Meek user makes connections to addresses registered with the high level domain name used such as Google. Meek uses these addresses as long as the Meek user is still connected to the Tor network using the same domain. However, an FTE user makes multiple connections resulting in traffic looking like HTTP.

B. *Transferred data and the number of connections*

While analyzing the amount of data sent and received compared to the number of connections that the pluggable transport users make, we observed that the relationship between these two variables shows which type of pluggable transport is used by the user. For example, a non-Tor user establishes multiple connections to download a file when using BitTorrent. The number of connections is high and at the same time the amount of data transferred is relatively high, too. In contrast, when using the pluggable transport, especially for Flashproxy, the number of connections is high but the amount of data is low compared to BitTorrent.

C. *Duration*

When the Tor user configures the Tor browser to use one of the supported pluggable transports over his/her connection with the Tor network, the duration of the connection is relatively high compared to a non-Tor user who is browsing websites. Even though if the non-Tor user browses only one website for a long time, this time is much less than a Tor user who is browsing multiple websites. In this case, the duration of the browsing of multiple websites associated with the Tor user points to one connection. The connection duration in most pluggable transports distinguishes the Tor users even if the pluggable transport obfuscates the traffic to look like random strings or HTTP traffic.

VII. CONCLUSION

Tor pluggable transports with their different forms provide evasions or resistance to censorships. Obfsproxy is the framework used by these pluggable transports to obfuscate the user connection to the Tor network. This obfuscation mainly concentrates on hiding the contents that make the connections to the Tor network recognizable. Consequently, using deep packet inspection cannot detect them as Tor. Pluggable transports successfully obfuscated Tor traffic to look like random or different forms of traffic. At the same time, this success to hide the content is not for free. The obfuscation in the pluggable transports changes the content shape distinct from Tor and therefor creates a fingerprint for the obfuscated pluggable transports. The results in this work show that pluggable transports' flows have their own unique fingerprints which make them recognizable.

Future work will explore the usage of other Tor traffic traces as well as other data mining algorithms for studying the best practices for feature selection and training set formations.

ACKNOWLEDGMENT

This research is partially supported by the Natural Science and Engineering Research Council of Canada (NSERC) grant, and is conducted as part of the Dalhousie NIMS Lab at http://projects.cs.dal.ca/projectx/. The first author would like to thank the Ministry of Higher Education in Saudi Arabia for his scholarship.

REFERENCES

[1] Tor Bridges. [Online]. Available: https://www.torproject.org/docs/bridges.html.en

[2] Z. Ling, J. Luo, W. Yu, M. Yang, and X. Fu, "Extensive analysis and large-scale empirical evaluation of tor bridge discovery," in INFOCOM, 2012 Proceedings IEEE, 2012.

[3] Tor Pluggable Transports. [Online]. Available: https://www.torproject.org/docs/pluggable-transports.html.en

[4] E. Hjelmvik, and W. John, "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] G. La Mantia, D. Rossi, A. Finamore, M. Mellia, and M. Meo, " Stochastic Packet Inspection for TCP Traffic, " in 2010 IEEE International Conference on Communications. IEEE, May 2010, pp. 1–6.

[6] J. Barker, P. Hannay, and P. Szewczyk, "Using traffic analysis to identify the second generation onion Router," in the 9th IFIP International Conference on embedded and ubiquitous computing, Melbourne, AUS, 2011, pp.72-78.

[7] A. Houmansadr, C. Brubaker, and V. Shmatikov, "The parrot is dead: Observing unobservable network communications," in Proc. of IEEE S&P, 2013.

[8] T. Wilde. Great firewall Tor probing circa. [Online]. Available: https://gist.github.com/twilde/da3c7a9af01d74cd7de7

[9] P. Winter, and S. Lindskog, "How the great firewall of China is blocking Tor," in Proceedings of the 2nd USENIX Workshop on Free and Open Communications on the Internet , USENIX Association,2012.

[10] D. Fifield, N. Hardison, J. Ellithrope, E. Stark, R. Dingledine, D. Boneh, and P. Porras, "Evading Censorship with Browser-Based Proxies," In PETS, 2012.

[11] P. Winter, T. Pulls, and J. Fuss. "ScrambleSuit: A Polymorphic Network Protocol to Circumvent Censorship," In Workshop on Privacy in the Electronic Society, Berlin, Germany, 2013. ACM.

[12] K. Dyer, S. Coull, T. Ristenpart and T. Shrimpton, "Protocol Misidentication Made Easy with Format-Transforming Encryption," ACM SIGSAC Conference on Computer and Commu- nication Security, CCS'13, pp. 61-72, ACM, 2013.

[13] Obfsproxy. [Online]. Availabe: https://www.torproject.org/projects/obfsproxy.html.en

[14] Meek. [Online]. Available: https://trac.torproject.org/projects/tor/wiki/doc/meek

[15] Obfs3. [Online]. Available: https://gitweb.torproject.org/pluggable-transports/obfsproxy.git/tree/doc/obfs3/obfs3-protocol-spec.txt

[16] TRANALYZER2 [Online]. Available: http://tranalyzer.com/

[17] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten,"The WEKA data mining software: an update," SIGKDD Explorations, vol. 11, no. 1, pp. 10-18, 20

[18] K. Shahbar, and A. N. Zincir-Heywood, "Benchmarking two techniques for Tor classification: Flow level and Circuit level classification,". in IEEE Symposium on Computational Intelligence in Cyber Security, 2014.