An Analysis of Tor Pluggable Transports Under Adversarial Conditions

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

***Abstract*— Tor Pluggable transports enable the users to overcome the adversaries which block access to the Tor network. Different pluggable transport systems use different mechanisms. Consequently, the adversaries adapt by using different approaches to identify Tor pluggable transport traffic. The deep packet inspection and the flow analysis are two of such approaches. To this end, we investigate how well pluggable transports can obfuscate user traffic under adversarial conditions. We represent the adversarial environments using the existing traffic analysis systems. Our results show that while some pluggable transports systems can hide the traffic well from adversaries, others cannot.**

Keywords—Network Traffic Analysis; Pluggable Transports; Adversaries

I. INTRODUCTION

The pluggable transports systems [16] [19] work to provide access to the Tor network in adversarial (censorship) environments. Most of the pluggable transports tools concentrate on hiding the content of the packets in a way that makes it hard for the adversaries when using deep packet inspection (DPI) to detect the connection to the bridges. But DPI is not the only method used to detect Tor traffic. The active probing and the flow analysis are some of the other popular methods used to detect Tor traffic.

In our previous work [8], we used the flow analysis approach to classify the Tor pluggable transports among other background traffic. In this research, we aim to extend our previous work in classifying Tor pluggable transports in terms of describing the proper features, the sufficient amount of data, and the effect of data collection for the flow analysis on Tor pluggable transports classification. We investigate the use of the flow analysis approach by adversaries against Tor pluggable transports to discover their presence. To this end, we will work on the following questions: (i) What is the detection and false alarm rate of such a flow analysis system could achieve to detect Tor traffic using pluggable transports? (ii) How can they profile the pluggable transports flows? (iii) What types of pluggable transports have the ability to evade flow exporters and up to what level? (iv) The pluggable transports currently supported by Tor designed to hide the payload content not the flow characteristics (except Scramblesuit). If this is the main goal of designing pluggable transports, how would such flows look in traffic traces?

The rest of this paper is organized as follows. Related work is reviewed in section II. The options available to the Tor users to connect to the Tor are summarized in section III. Section IV details the Tor blocking mechanisms and their corresponding countermeasures by Tor pluggable transports. The flow analysis approach and the tools used in this research are presented in section V. Data collection is discussed in section VI. Results are presented in section VII. Finally, the conclusions are drawn and the future work is discussed in section VIII.

II. RELATED WORK

Tor pluggable transports use the obfuscation concept to resist the blockage of the Tor service. There are several researches focused on the analysis and the identification of obfuscation techniques.

Hjelmvik and John [1] used statistical analysis to fingerprint the traffic of obfuscated protocols. The authors proposed Statistical Protocol Identification (SPID) algorithm to identify applications that use obfuscation such as Skype and BitTorrent. Barker et al. [2] used machine learning algorithms to distinguish between HTTPS, HTTP over Tor, and HTTPS over Tor. Wiley [3] used a Bayesian model to identify obfuscated protocols (SSL, Obfsproxy, and Dust protocol).

SkypeMorph [4] and StegoTorus [5] are two Tor Pluggable transports that are proposed but not implemented (yet). SkypeMorph aims to shape the connection to Tor bridges to make it look like a Skype call. StegoTorus uses blocks of data and padding to change the characteristics of the client connections to Tor. Houmansadr et al. [6] showed that CensorSpoofer, SkypeMorph, and StegoTorus could be distinguished from the protocol they tried to mimic in the simulations they performed. The authors observed that SkypeMorph mimicked the datagram behavior of Skype call but it could not mimic the TCP control channel that accompanies the Skype call. They concluded that this made SkypeMorph detectable compared to a Skype call. As for,

StegoTorus, they argued that it is not similar to any known web server and could make StegoTorus recognizable among the known HTTP web servers. Furthermore, they showed that for such a tool to successfully mimic any protocol, it had to meet the following requirements: (i) complete mimicking of the target protocol and its accompanied traffic, (ii) the right response for errors, and (iii) mimicking the content and behavior of the protocol.

In our previous work [7], we compared two different techniques to classify the activities of the Tor users into one of three classes. These were Web browsing, BitTorrent, and Videos. The first technique was circuit level classification and the second one was the flow level classification. The circuit level classification is based on extracting information form the Tor application itself which requires access to the Tor node. The flow level classification is based on the communication between the user and the entry node. The classification depends on a prior knowledge that the user is using Tor. The main goal was to classify the type of traffic within a known Tor traffic. Furthermore, in [8], we aimed to study Tor obfuscation techniques and differentiate such Tor traffic from background traffic. In those experiments, the data we used were collected from a real network environment for all the available Tor pluggable transports. Based on the high accuracy results, in this paper, we aim to extend our work to achieve better understanding of Tor pluggable transport identification under different adversarial conditions.

III. CONNECTION TO THE TOR NETWORK

Tor provides several options for the users to connect to the Tor network. The following details these options.

A. Process of Connecting to the Tor Netwok

When the Tor user connects to the Tor network, the user starts by creating virtual circuits with the first node then the second node and extends the circuit to the third node. The IP addresses of all Tor nodes are not hidden. It is possible to download the list of all available nodes through the Internet. There are websites that provide all the available nodes and update them periodically. The list could also be downloaded from the directory authority itself. The IP address of the first node is the only IP address from the three nodes that appears to an observer during the user connection to the Tor network. The IP addresses of the second and third node do not show when analyzing the user connection to the Tor network due to the encryption. Preventing the user from connecting to the first node by blocking the IP address of all nodes is enough to block the Tor network service for the user.

B. Using a bridge

Even though the Tor network aims to hide the activities of its users on the Internet and their identities, it does not hide that the user is connecting to the Tor network. The IP addresses of all the Tor nodes are publically available. In some countries, these addresses are used to block Tor. Therefore, Tor provides an option for the user to connect to the Tor network using a bridge [14]. A bridge works as an entry node to the Tor network. The bridge is a normal node but its IP address is not announced in the public list. The user can get the IP address of the bridge by sending an email to the Tor network. There is a limitation on the number of IP addresses a user can get daily to prevent exposing the bridges from been blocked. The IP address of a bridge could also be found on the bridge database website with the same limitation on the number of IP addresses that can be used per day.

C. Connecting to the Tor Network by Using Pluggable Transport

Using a bridge to hide that the user is connecting to the Tor network is a good solution but it is not optimal. The method that is used to provide the users with the IP addresses of the bridges could be used by an adversary to accumulatively collect the IP address of all bridges over time. Tor provides the pluggable transport as another way that helps the users to connect to the Tor network by obfuscating their traffic in some way. The pluggable transports do not count on using unknown IP addresses; instead they aim to change how the user connects to the Tor network.

IV. DISCOVERY AND BLOCKAGE OF THE TOR NETWORK.

As discussed earlier, the Tor network could be blocked by different methods such as blocking the IP address of the nodes, discovering and blocking the Tor bridges, using active probing, performing DPI, implementing whitelisting, and using flow analysis. For example, the IP addresses of the Tor nodes and directory authorities are used to block Tor service by blocking these IP addresses [23]. Moreover, Ling et al. [15] used two different methods to reveal the IP addresses of Tor bridges. The first method was using bulk email accounts to request IP addresses of many bridges daily. Since Tor allows only three IP addresses daily, they created 2000 email accounts using different tools (iMacros, PlanetLab, the Tor network itself) to automate retrieving the addresses of the bridges by email and overcome the limitation of IP addresses. In addition to using the email to request to collect the IP addresses of the bridges, 1000 planeLab nodes were used to request the addresses of the bridges through the web.

The second method seems to be more practical and effective than the bulk email accounts. Tor has its policy to classify routers inside the Tor network as entry, middle and exit routers. This policy includes but not limited to weighting the bandwidth of each router, measuring their uptime, averaging the bandwidth available in the network, collecting reports about suspicious routers and applying the exit routers policy set by the router itself. By manipulating these factors, Ling et al. aimed to insert one router in the Tor network and let the directory server to select it as a middle router. Whenever a user tries to connect to the Tor network using a bridge, if the connection comes through the middle router to establish a connection to a bridge router, then the address of the bridge is obtained through the comparison of the announced entry routers and the address passed through the middle router. If the address of the router is not in the public list of all routers then this address is a bridge address. Active probing is another method used to find Tor bridges. It is used to send a connect

request to the IP address that is obtained from intercepting TLS connections (Tor uses TLS), when a reply is obtained. Then, this confirms that this IP address belongs to a bridge. Wilde [17] investigated how the Great Firewall of China (GFW) discovers connection to bridges from China. The connection from the user to the bridge must take place first. Then if inspecting the packets shows that there is an SSL connection, it establishes a connection to this IP address. If the connection is a success, then this is a bridge IP. The IP address and its ports are blocked. The result of Wilde's work was used by Winter & Lindskog [18] to understand more about how active probing is used to block Tor. Whitelisting is another method that could be used to list the allowed traffic and block any traffic that does not match the permitted type of traffic in the list.

Tor pluggable transports is an effort developed by the Tor community to evade the aforementioned blocking techniques. To this end, Obfs3 [9] is a pluggable transport that changes the TLS to random strings. It adds an extra layer of encryption to the TLS used by Tor. Obfs3 changes only the shape of the TLS; it does not change the packet length or the timing of the packets. Another Tor pluggable transport is Scramblesuit [13]. One of the design goals of the Scramblesuit pluggable transport is to resist the active probing by using a ticket and password to connect to the server. Flashproxy [10] is a pluggable transport that can avoid blocking by IP addresses. The Flashproxy uses the browser of the visitors of "Flashproxy supported websites" to connect the Tor user to the Tor nodes. Therefore, the IP addresses of the users keep changing based on the IP addresses of the visitors of the Flashproxy supported websites. Format-transforming encryption (FTE) [12] is a pluggable transport that changes the format of a ciphertext to another form that matches a regex. For example, FTE changes the format of the Tor traffic to an HTTP-like regex that matches the regex of HTTP traffic. Last but not the least, Meek [11] encapsulates Tor messages into an HTTPS request by using "domain fronting". It uses Google, Amazon, and Azure as a domain fronting to send Tor messages.

Many of the known cases where Tor is blocked were mainly implemented using a DPI. Therefore, most of the pluggable transports aim to evade the DPI technique. On the other hand, to the best of our knowledge, the flow analysis approach has not been evaluated thoroughly by the Tor community in terms of its detection and false alarm rates. Thus, the abilities of an adversary who might be using a flow analysis approach are not well-known. In short, in this paper, we investigate the effectiveness of the adversary using the flow analysis approach to identify the obfuscated Tor traffic.

V. TOR CLASSIFICATION ALGORITHMS AND TOOLS

We evaluated different classification algorithms and approaches in our previous work [7], to identify the type of applications used on the Tor network. C4.5, Random Forest, Naïve Bayes, and Bayes Net were algorithms evaluated. Based on the results, C4.5 was the preferred algorithm to classify Tor. Moreover, as a potential flow exporter, we evaluated Tranalyzer [20] and Tcptrace [22]. Our results showed that Tranalyzer based traffic analysis system performed better than the Tcptrace based traffic analysis system. Therefore, in this paper, we use Tranalyzer as the flow exporter and the C4.5 decision tree classifier to perform the traffic analysis.

Flow exporter tools use the following five tuple to define a flow: The source IP address, the destination IP address, the source port, the destination port, and the protocol. Once the flows are exported, we removed this information from our data sets to ensure that the classification process is not biased using this information. Not using the ports numbers in the analysis, also eliminates the bias in terms of linking a port number to an application. This is important since many applications use dynamic port numbers on the Internet. Furthermore, for most of the Tor pluggable transports, the port number is configurable by the server and sometimes by the client side. Pluggable transports can be configured even to use well-known ports such as ports 80 or 443.

Tranalyzer, which is the flow exporter that is used in this work, extends the Netflow. Tranalyzer supports the traffic analysis from (a single or multiple) pcap file(s) or directly from an Ethernet interface. Tranalyzer configuration allows the user to include the required modules (plugins) that meet the user requirements. The output of Tranalyzer changes according to the chosen plugins. Some examples of the 65 Tranalyzer features are: Number of packets been transmitted, Number of packets been received, Minimum packet length, Maximum packet length, Average packet size, and Mean inter-arrival time. The complete list of features could be found in [20].

The results in the following section are calculated using the following performance measurements: The first metric "Accuracy" is defined as the summation of True Positive (TP) and True Negative (TN) values divided by the total number of instances (N) as shown in Eq. (1). For example, when measuring the accuracy of the classification for the HTTP traffic, TP is the total number of correctly classified instances as HTTP. TN is the total number of correctly classified instances as non-HTTP. As shown in Eq. (2), Precision is the ratio of TP divided by the summation of TP and False Positive (FP). For example, If a HTTPS instance is classified as HTTP instance, then this is considered as FP. The opposite is when the classifier classifies an instance as a non-HTTP while it is a HTTP, then this is a False Negative (FN). Eq. (3) defines the Recall as the division of TP over the summation of TP and FN. The relation between precision and recall is shown in Eq. (4) as an F-measure. It calculates the overall performance of the classifier.

$$\text{Accuracy} = \frac{\text{True Positive (TP)} + \text{True Negative(TN)}}{\text{Number of Instances(N)}} \quad (1)$$

$$\text{Precision} = \frac{\text{True Positive (TP)}}{\text{True Positive (TP)} + \text{False Positive(FP)}} \quad (2)$$

$$\text{Recall} = \frac{\text{True Positive (TP)}}{\text{True Positive (TP)} + \text{False Negative (FN)}} \quad (3)$$

$$\text{F-Measure} = \frac{2 * Precision * Recall}{Precision + Recall} \quad (4)$$

TABLE I. TOTAL NUMBER OF FLOWS OF THE BACKGROUND TRAFFIC

	Background Traffic				
Type	***HTTP***	***HTTPS***	***SSH***	***BT***	***Encrypted BT***
Number of flows	182725	8058	54214	116440	198302
Total	**559739**				

TABLE II. TOTAL NUMBER OF FLOWS OF THE PLUGGABLE TRANSPORTS TRAFFIC

	Tor Pluggable Transports Traffic				
Type	***Obfs3***	***FTE***	***Scramblesuit***	***Meek***	***Flashproxy***
Number of flows	15356	106549	16953	43152	172331
Total	**354341**				

VI. DATA COLLECTION

The technique behind each one of the Tor pluggable transports is different based on the main goal of the pluggable transport system. Some of the pluggable transports are designed to be used with Tor and non-Tor systems. This generates different flow behaviors. Therefore, the way we collected the data from one pluggable transport to another also changes based on their behavior. For example, Flashproxy assigns multiple IP addresses (IPs of the website visitors) to the Tor user. On the other hand, the connection between the Obfs3 server and the Tor user employs only one IP address. This may be the IP address of one of the default servers or an Obfs3 server that the user chooses to connect to. The number of flows in the Flashproxy case is much higher than the Obfs3 case even if both connections stay active for the same duration of time and have the same Internet activities. So in our experiments, we connect to multiple Obfs3 servers to generate sufficient amount of flows.

Given that pluggable transports hide the Tor traffic using different protocols, we make sure to include HTTP, HTTPS, SSH, BitTorrent (BT), and Encrypted BitTorrent (BT) traffic in our evaluations as the background traffic. Table I and Table II present the number of flows for the background traffic and the pluggable transports, respectively. All of the data generated and captured in this work is made publicly available for the research community at large [25].

VII. RESULTS

We generated and captured different datasets from the flows of the pluggable transports and the other traffic (background) on the network. The first dataset contains all the instances (914080 instances) with all the features (65 features), and it is labeled (groundtruth) using 10-classes. The experiments on this dataset are performed by splitting the instances as the training and the testing instances.

TABLE III. RESULTS PER CLASS ON THE FIRST DATASET USINGTHE SPLITTING TECHNIQUE

	Class	TP Rate %	FP Rate %	Pre-cision %	Recall %	F-Measure %
Background Traffic	***HTTP***	99	0.1	99	99	99
	HTTPS	94	0.1	94	94	94
	SSH	99	0	99	99	99
	BT	94	2.6	80	94	88
	BTecr	89	0.9	96	89	92
Pluggable Transports Traffic	***FTE***	99	0.1	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%***					

The second dataset contains all the instances and the labels of the first dataset, but in this case we used a smaller number of features, namely three selected features, to represent the data. The third dataset contains all the instances and the features of the first dataset, but in this case, we labeled the dataset using only two classes, namely Tor and non-Tor.

A. Splitting the data into 66% Training / 34% Testing

We split the data into a training set (2/3) and a testing set (1/3). The training set is 66% of the whole data. The testing set is the remaining instances. Thus, the goal in using this approach is to investigate whether the classifier would be able to learn the properties of each of the 10 classes on the training set only. Then we test, how well it learned (if at all) on the unseen (not seen during the training) test data set. When the C4.5 traffic classification system is used on this data set, the percentage of correctly classified instances is 97%. The performance measures are shown in Table III.

B. 10-Fold Cross Validation

In our previous paper [8], we evaluated our classifiers using only the 10-fold cross validation technique on the first dataset. We included the results of this approach from the previous paper in Table IV.

Fig. 1 shows the F-Measure of the 10-fold cross validation technique and the split technique for the background traffic and the pluggable transports traffic. The lower F-measure values in these experiments indicate that the BT, the encrypted BT, the HTTPS and the Scramblesuit traffic flows are more difficult to identify compared to the other applications. The traffic flows of these applications also have differences between the 10-fold cross validation results and the split results. The traffic of the other applications showed consistent results between the 10-fold and the split technique. This seems to indicate the training set when using the split technique for these applications contains instances that did not appear on the testing test and vice versa.

TABLE IV. RESULTS PER CLASS ON THE FIRST DATASET USING THE 10-FOLD CROSS VALIDATION TECHNIQUE

	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%***					

TABLE V. RESULTS USING ONLY THREE FEATURES

	Class	TP Rate %	FP Rate %	Pre-cision %	Recall %	F-Measure %
Background Traffic	***HTTP***	97.4	0.8	96.9	97.4	97.1
	HTTPS	79.1	0.1	85.2	79.1	82
	SSH	98.7	0	99.9	98.7	99.3
	BT	82	2.7	81.7	82	81.9
	BTecr	86.3	3	88.8	86.3	87.5
Pluggable Transports Traffic	***FTE***	97.7	0.3	97.4	97.7	97.6
	Scramble suit	84.6	0.1	92	84.6	88.1
	Meek	95	0.4	91.7	95	93.3
	Flash proxy	97.8	1.2	95	97.8	96.4
	Obfs3	98.3	0	98.8	98.3	98.6
Overall Correctly Classified Instances	***93%***					

C. Feature Selection

The number of features that are used to represent the traffic in the above experiments is 65. These are all the relevant features of Tranalyzer for our purpose, i.e. the analysis of Tor pluggable transports flows. The non-relevant features such as ICMP and VLAN are excluded from the flow analysis. But this large number of features, when combined with the number of instances (914080) we have, makes the classification computationally costly. So we used the feature selection technique, Ranker from WEKA [21] to reduce the number of features. Ranker is a search method to arrange the features from the most important to the least important. Based on the results of Ranker, we picked the most important three features: the Duration, the Number of Bytes Sent, and the Maximum Packet Size. The performance of our classifiers using only these three features is shown in Table V.

TABLE VI. RESULTS FOR BINARY CLASSIFICATION

Class	TP Rate %	FP Rate %	Precision %	Recall %	F-Measure %
Tor	99.7	0.3	99.5	99.7	99.6
NonTor	99.7	0.3	99.8	99.7	99.8
Overall Correctly Classified Instances	***99.7%***				

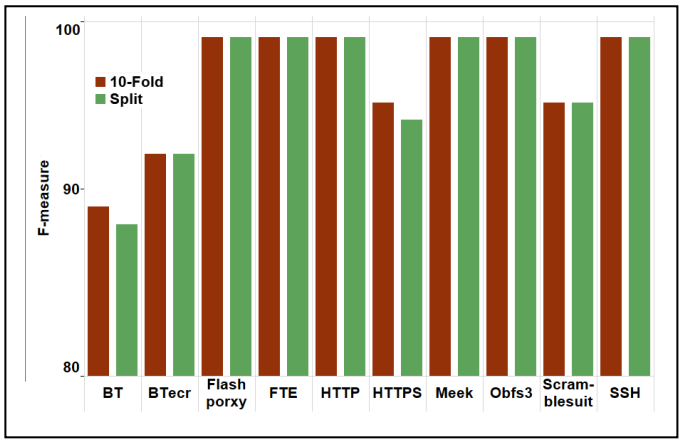

Fig. 1 Comparison of F-Measure values between the 10-fold cross validation technique and the splitting technique on the first dataset.

D. Binary Classificaiton

In this experiment, we labeled all the pluggable transports in our data set as Tor and labeled all the other traffic as non-Tor. This brings up the total number of Tor traffic to 354341 instances and for the non-Tor traffic to 559739 instances. In this case, the percentage of overall correctly classified instances is increased to almost 100%. The results of this experiment are presented in Table VI.

In this case, Fig. 2, all the background instances are put together into one group and all the pluggable transports instances are put in another group. The x-axis represents the duration and the y-axis represents the average packet size for every flow in each group. The number of instances is large and sampled from all the data sets. The duration is limited to five minutes. The average packet size for an individual flow of pluggable transports lies in the middle area of the graph with few outliers. The average packet size of the other (background) traffic is scattered all over the graph. This relationship between the data points indicates that the average value of the data transmitted by Tor can be used to distinguish Tor from non-Tor traffic. It seems like the tools that change the packet length such as Scramblesuit do not change the average amount of the data transmitted in a way that makes it indistinguishable. We believe that this is because the amount of padding used to change the length of the packets is small. This in return, does not completely change the total amount of Tor data transferred compared to non-Tor data. The use of this phenomena is important. Because hiding the 512 bytes cells by padding still does not change the total amount of data transferred.

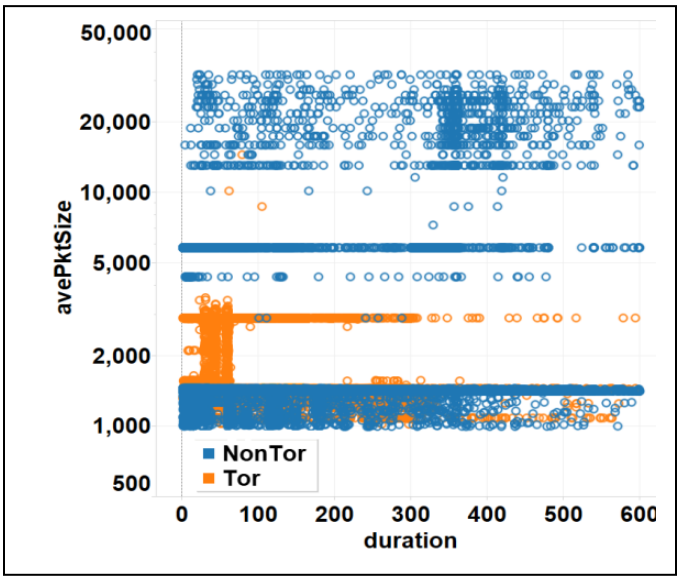

Fig.2 The distribution of average packet size

E. Size of the Training Set for Reasonable Performance

To explore the question of how many instances are required to achieve a reasonable performance; we decreased the training set size to 10% of the two classes data set (Tor and non-Tor). Again, we used all the 65 features to represent the traffic. In this case, we were able to classify 99.2% of the instances correctly. Furthermore, with using only the 5% of the data set as the training set, the performance only dropped down to 98.9%. Finally, using only 1% of the data set as the training set (914 instances), the correctly classified instances were 98.1%. This means that out of the 904939 classified instances (flows), 888185 were correctly classified.

VIII. CONCLUSION

The results above show that an attacker who has the means to perform flow analysis against Tor could achieve a very high performance in detecting the Tor pluggable transports under all the conditions we evaluated.

Section VII-C shows that when the feature selection is used for the flow analysis of the pluggable transports, One can identify the important features that describe the pluggable transport behavior. In our experiments, the duration, the number of bytes sent, and the maximum packet size seem to be the most important features.

The pluggable transports are designed to hide or obfuscate the content of the Tor connection not the flow. Thus, the flow analysis could identify Tor traffic even with the existence of such obfuscation techniques. Among these techniques (tools), even when Scramblesuit has the ability to change the distribution of the packet length and the inter arrival time of the traffic, the flow analysis could still profile different Tor traffic behaviors with an 85% true positive rate (just by using the aforementioned three features). If more features are used, then the detection rate goes up to 98%. Additionally, under other obfuscation techniques, Tor behavior classification could even reach up to 98% true positive rate.

Having said this, the detection rate might change based on the background traffic characteristics. For example, FTE obfuscates by making the regex of the Tor encrypted traffic look like the regex of HTTPS traffic. This makes the HTTPS traffic an important factor in the training data set.

Furthermore, the feature selection indicates that the packet size, the number of bytes sent, and the maximum packet size are the three main features that profile the pluggable transports. In fact, this observation is also consistent with how Tor works. For example, Obfs3 does not change the packet size nor the inter arrival time. This makes the packet size an important feature that profiles the Obfs3 traffic. Obfuscating the TLS handshake has nothing to do with the amount of transferred data between the user and the pluggable transport server. The duration of the connection is one of the features that could profile Tor traffic with or without the pluggable transports. Regular connections stay active for a shorter time than the duration that a Tor user connection does.

Moreover, based on our observations on the network traffic, Flashproxy changes the connection to the user based on the IP addresses of the FlashProxy supported websites. This causes the importance of the duration in the detection of FlashProxy to become less compared to the importance of the packet size. Scramblesuit changes the packet size (to a certain level) but the duration is still a factor. In summary, our evaluations seem to indicate that pluggable transports designed mainly to evade DPI. However, they need to consider the flow analysis in their design to improve the obfuscation of the Tor traffic.

Future work will investigate other anonymity services using our proposed approach. In doing so, we aim to identify the limitations of of these systems against the existing data driven traffic analysis techniques.

REFERENCES

[1] 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.

[2] 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.

[3] B. Wiely, “Dust: A Blocking-Resistant Internet Transport Protocol,” Technical report, University of Texas at Austin, 2011.

[4] H. Moghaddam, B. Li, M. Derakhshani, and I. Goldberg, “SkypeMorph: protocol obfuscation for Tor bridges,” in Proceedings of the 2012 ACM conference on Computer and communications security, 2012.

[5] 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 Proceedings of the 2012 ACM conference on Computer and communications security, 2012.

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

[7] 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.

[8] K.Shahbar, and A. N. Zincir-Heywood, "Traffic Flow Analysis of Tor Pluggable Transports," in the 11th International Conference on Network and Service Management, 2015.

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

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

[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] 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.

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

[15] 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.

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

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

[18] 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.

[19] Obfsproxy. [Online]. Available: https://www.torproject.org/projects/obfsproxy.html.en

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

[21] 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

[22] TCPTRACE [Online]. Available: http://www.tcptrace.org/

[23] A. Lewman, "Tor partially blocked in China". [Online]. Available: https://blog.torproject.org/blog/tor-partially-blocked-china

[24] Tor blog, "Updae on Internet censorship in Iran" [Online]. Available: https://blog.torproject.org/blog/update-internet-censorship-i

[25] Anon17: Anonymity Network Dataset .[Online]. Available: https://web.cs.dal.ca/~shahbar/dataset/anon17