Traffic Analysis Attacks on Tor : A Survey

Lamiaa Basyoni[1], Noora Fetais[2], Aiman Erbad[3], Amr Mohamed[4], and Mohsen Guizani[5]

[1,2]Kindi Center for Computing Research- Qatar University
[3,4,5]Computer Science and Engineering Department - Qatar University
[1]lamiaa@qu.edu.qa, [2]n.almarri@qu.edu.qa, [3]aerbad@qu.edu.qa, [4]amrm@qu.edu.qa, [5]mguizani@qu.edu.qa

***Abstract*—The Tor anonymity network is one of the most popular and widely used tools to protect the privacy of online users. Tor provides defenses against multiple adversarial activities aiming to identify or trace the users. Traffic analysis is a very strong tool that can be used for internet surveillance. Traffic analysis attacks against Tor's anonymity network has been known as an open question in research. Moreover, the low-latency feature Tor tries to provide to its users imposes limitations in defending against traffic analysis attacks. In our study, we examine traffic analysis attacks from the perspective of the adopted adversary model and how much it fits within Tor's threat model. The purpose of this study is to evaluate how practical these attacks are on real-time Tor network.**

***Index Terms*—Anonymity Network, Tor, Security, Traffic Analysis, Attacks.**

I. Introduction

Internet communications are becoming subject to all kinds of surveillance and attacks threatening the security and privacy of web users. Traffic surveillance in some cases is implemented by governmental entities to silence opposing voices, or even to track people down. Moreover, clients while surfing the Internet they unintentionally reveal their identities by leaving traces of their online activities. To serve the purpose of preserving users' online privacy and security, Tor anonymity networks aim to hide the identity of the users by distributing the traffic among multiple relays. Since its introduction in 2003, Tor has gained more popularity, currently serving hundreds of thousands of users every day. The anonymity provided by Tor depends on separating the originator of the traffic from its destination by re-routing the traffic through a chosen path of multiple relays and encrypt the entire traffic, making it harder for any observer to link the origin and destination. Moreover, Tor provides anonymity to the server-side through hidden services which can be connected to through an Onion address given by Tor. Most of the attacks aiming to de-anonymize Tor's clients depend on analyzing the observed traffic entering, exiting, or passing through Tor's network. One of the most commonly used methods for attacking the anonymity network is traffic analysis. Traffic analysis is the process of examining the traffic flow with the intention of inferring certain information from it. Due to the low-latency characteristic of Tor's anonymity network, Tor does not provide sufficient protection against traffic analysis attacks. [1] [2] [3] [4]

In this paper we, are exploring the traffic analysis attacks on anonymous communication networks, specifically Tor. We study the adversary models of traffic analysis attacks on Tor in terms of its practicality and matching with the threat model that Tor actually is designed to defend against. The contribution of our study is to highlight the relation between the traffic analysis attacks types and the adopted adversary model, the study is also a first step towards evaluating the practicality of launching these types of traffic analysis attacks on real-time Tor network.

II. Background

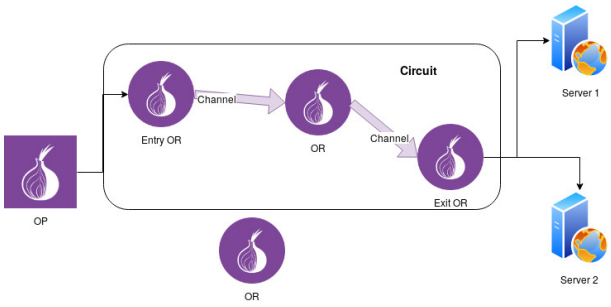

Figure 1: Tor network

The Tor network consists of multiple relays called *onion routers (ORs)*. A *descriptor* of each OR is constructed containing the OR's keys and address. The descriptors are then sent to the *authority directory* relays. When a client tries to use the Tor network in order to communicate with a server, it connects through a proxy called *Onion Proxy (OP)*. The OP establishes a path to the destination address called a *circuit* typically consists of three ORs. To create a circuit the OP first contacts the authority directory and chooses an OR to be its entry point (*entry guard*). The entry guard then *extends* the circuit to the next hop, reaching the *exit guard* which is the last hop on the circuit. The exit guard then connects to the destination, figure 1. Each two communicating ORs maintain a single TLS/TCP connection called *channel*. Traffic flow from one client to one destination is logically viewed as a *stream*, each stream is mapped to a circuit, and multiple circuits are multiplexed over the same channel. Circuit creation in Tor is done in an incremental way, the OP chooses an entry OR, sends a *create* command to it, and exchange keys. Once the entry OR replay with *created*, the OP *extend* the circuit to the next hop. The full sequence of circuit creation is explained in

978-1-7281-4821-2/20/$31.00 ©2020 IEEE

figure 2. Router selection is done by using a weight depending on the available bandwidth.

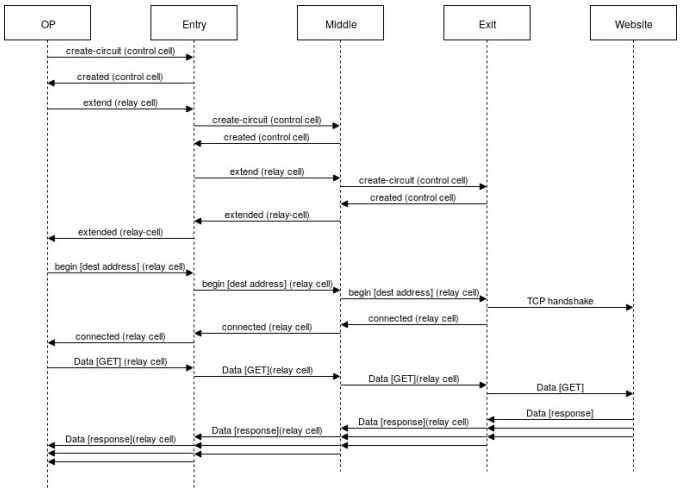

Figure 2: Tor circuit creation

Tor uses fixed-size packets called *cells* of size 512 bytes. There are two types of cells, the *control cells* are conveying command relating to building, extending, and destroying circuits and flow control commands, the structure of this type is shown in figure 3. The second type is the *data cells* structured as shown in figure 4, and they carry the end-to-end communication messages between the client and the server. The fields CircID and Command are not encrypted and are used at each OR on the circuit to route the cell to the corresponding circuit queue. The rest of the cell is encrypted and can only be processed at the exit OR in order to complete the connection. [5]

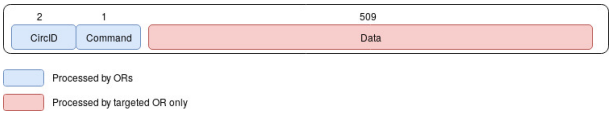

Figure 3: Structure of control cell

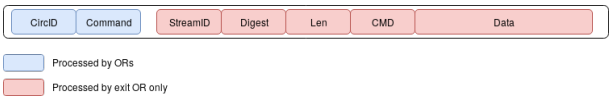

Figure 4: Structure of data cell

Hidden services were introduced to Tor in 2004, to provide the anonymity feature not only to the client-side but to the server-side as well. In order to be reachable by clients, the service provider (service OP) starts by generating a hidden service descriptor, and follows these steps:

- The service OP chooses some relays randomly and builds a circuit with each one of them. These relays are now serving as *Introduction Points(IPs)* to this service. The selected relays share their public key with the onion service.
- The service creates a *service descriptor* containing the public keys of the IPs signed by its own private key. This descriptor is then shared and distributed to be found by any user tries to find this particular service using its *.onion* address.
- Once the client download the service descriptor it selects a random set of relays to act as *rendezvous points* and start building circuits to these relays using a one-time secret.
- The client then forms an *introduce* message containing the address of the rendezvous point and the one-time secrete, and encrypt the message using the onion service's public key. The client sends this introduce message to the introduction point asking to pass it to the onion service.
- When the onion service receives the introduce message, it decrypts it using its private key, read the rendezvous point address and its one-time secret. The onion service now forms another message called *rendezvous message* containing the one-time secret and send it to the rendezvous point over a circuit that it builds.
- The rendezvous point informs the client that the connection has been established to the service, and the client is now ready to communicate with the service.

[6]

III. Attacking Tor

A. Tor's Threat Model

In low-latency anonymity networks such as Tor, an adversary is generally aiming to confirm the source and destination of communication. The current design of Tor's network assumes the absence of a global adversary that is able to monitor both ends of the communication, entry, and exit guards, and does not provide anonymity against this type of adversaries. Instead, Tor's threat model assumes an adversary that can observe only a fraction of the communication, and is able to control only a fraction of Tor nodes, either by running his own ORs or compromising an already running ORs [7]. Based on the proposed threat model, attacks can be categorized according to how practical their model is, in terms of the assumptions made and the required resources to enabling the attack. In the analysis of the security of the onion router provided by Syverson, *et al* [8], the probability of facing an adversary compromising either the first node or the last node on the route is the same and is equal to : (the number of compromised nodes (c)) / (the total active nodes in the system (n)). However, an adversary compromising the first *and* the last nodes on the route might exist with probability = c^2/n^2. These probabilities are valid for a certain time duration, routes are dynamic in Tor which means that after a specific amount of time, if the path is not used to exchange traffic it will be automatically torn down and another path will be established for future activities.

B. Tor's Defenses

Tor encrypts the client's traffic using multiple keys and implement perfect forward secrecy, hence, for an attacker to

attempt to compromise the encryption keys he has not only to learn the OR's TLS session key but also the circuit session key, and due to the periodic rotation of circuit key the window available to launch such attacks is very limited. Tor also forces circuit lifetime limit after which the ORs will erase all information that an adversary, who compromises an OR on the circuit needs to carry the attacks further and compromise more nodes. In addition, Tor provides solid defenses against tagging attacks, and replay attacks.

On the other hand, traffic confirmation attacks are generally out of the scope of Tor's design, therefore, Tor provides minimal protection against adversaries aiming to correlate end-to-end timing and packet size. Website fingerprinting is another type of potentially effective attacks on Tor's network, in which an attacker collects *"fingerprint"* of highly targeted websites containing the access patterns and tries to map observed traffic to these fingerprints [1]. A defense was developed by Tor against fingerprinting attack by enabling HTTP pipe-lining, however, later research proved that the proposed defense was not effective to prevent fingerprinting attacks [9] [10].

IV. Traffic Analysis Attacks

The main focus of traffic analysis is to develop algorithms and procedures to enable observing, analyzing, evaluating, and controlling communication. In the context of anonymity networks, such as Tor, traffic analysis is used to reveal the identity of Tor's users, or understand their network behaviour. Traffic analysis attacks success depends on how accurate the adversary information. The higher network coverage of the adversary model the more probable the traffic monitored will be accurate. However, the design of a threat model should be aware of impractical assumptions made about the duration of observation as well as the percentage of network coverage. In the following, we will discuss traffic analysis attacks from the perspective of threat models. [11] [12]

A. Global Adversary Model

Considering the previous discussion of Tor's threat model, the *global adversary* is out of Tor's scope of defenses. Nevertheless, there are proposed traffic analysis attacks on Tor's network that go for the assumption that the adversary is monitoring both ends of the communication [13] [14] [15] [16] [17]. In the following will be exploring the threat models of these attacks and the requirements of launching them.

The traffic correlation attack (*DeepCorr*) proposed by Nasr, *et al* [13], is a traffic correlation attack implemented by using a deep learning technique with a learning function developed mainly for Tor's network. In their setup, data was collected by observing the traffic from the Tor client to the entry guard, as well as the traffic exiting the Tor network to the destination. The flow features used in DeepCorr, namely, the *inter-packet delay* and *packet size* are introduced as raw features to the deep learning method to derive complex features which is then used in the correlation function. Although DeepCorr shows considerable improvement in the correlation accuracy compared to other traffic correlation techniques [18] [19], the accuracy of DeepCorr was drastically reduced over time, and the researchers recommended retraining the system once every month. The researchers suggested obfuscating the entire Tor traffic or at least the traffic from the OP to the entry as a possible countermeasures for this attack.

Chakravarty, *et al* [15], used NetFlow data records to launch an active traffic analysis attack on Tor. The threat model assumed is that the attacker is capable of observing the NetFlow records of the routers around targeted Tor ORs. The attacker is also assumed to be able to select a particular anonymous connection to track. Two scenarios were proposed for the active attacker. The first one has the active attacker controlling a web-server and using it to inject a variation pattern in the victim traffic. The second scenario is for a malicious client aiming to de-anonymize a hidden service and injecting the traffic pattern from the OP side. The next step after transferring the traffic with the injected pattern is for the attacker to collect the NetFlow records at the two ends, *client-to-entry* and *exit-to-server*, and then compute the correlation between them. In the attack implementation, the researchers assumed the traffic was a download of large files to maintain 5 to 7 minutes of traffic to be captured. The correlation accuracy of the attack on a real Tor network was not sufficient due to some parameters such as congestion.

Under the same assumption of a global adversary, Song, *et al* [14], proposed a traffic analysis attack using a *support vector machine (SVM)* method for machine learning. The proposed model of adversary assumes that the adversary is able to capture the flow from OP to entry guard and the flow from exit guard to destination. The featured used to associate the traffic are the time and the stream size(at this particular time). The correlation accuracy of the attack was decreased over time.

B. Capturing Entry Flow

This threat model falls within the scope of Tor's defenses, where the adversary is only able to monitor part of the traffic. *Entry flow* is the traffic flow from the client's OP to the entry guard. Observing the entry flow is a realistic assumption and can be done in real life by many entities, such as governments. In their research [20], Gilad and Herzberg introduced a traffic analysis attack based on TCP side-channel information. The attack was proposed in two scenarios, one of them concerning a plain TCP connection between the client and the server. The second scenario considers an anonymized connection over Tor's network. The described attack makes use of active traffic shaping. using spoofing traffic the attacker changes the communication rate to limit the number of available exit relays for the client to choose from. The attacker actively changes the pattern of the traffic sent from the server to the exit node, then tries to detect if the entry flow captured is equally affected. The implementation and evaluation of the attack on Tor was very limited.

A similar attack was proposed by Arp, *et al* [21], in their *Torben*. Torben designs side-channel attack based on the fact that Tor cannot hide the traffic information very well, which means that for web browsing, the size of the HTTP

request and response actually affects Tor's encrypted traffic. The scenario of launching Torben involves an active attacker controlling a server that the victim client is interested in. The attacker introduces a particular marker in the traffic delivered to the victim through Tor's network. The marker Torben used is a number of request-response pairs of large size to be distinguishable from regular traffic, and then split them into quad-bits, and encode each one separately. The attacker then observes the entry flow to detect the presence of the marker. The live experiments showed accuracy of 91%, however, the experiment was very limited and only explored 34 marked web pages. The attack also does not hide the marker very well, which makes it easier to detect it and develop a defense against this type of attack.

Identifying Tor traffic and separate it from regular traffic is another type of traffic analysis attacks, it can be used to block Tor users from being able to access the web. In their research Lashkari, *et al* [22], showed that using only time-based features they were able to characterize Tor's traffic and identify it within a flow. The collected data was obtained from Tor's entry flow (flow between client and entry guard). In order to label the collected data for training the machine learning model, the researchers used a controlled environment in which one application was used at a time (Browsing, E-mail, Chat, File transfer, and streaming). They used the inter-arrival time, flow bit-per-seconds, and flow duration as features.
A similar method was proposed by He, *et al* [23], using burst volumes and directions as model features. The attacker in this threat model captures the entry flow for each application type. The target applications to be identified in the flow were P2P, FTP, IM, and Web browsing. Lashkari's classification method covered more internet protocols compared to He's method, and with accuracy reaching 99% while He's method was only 92% accurate.

C. Compromising Tor's relays

One of the widely adopted threat models by many attacks assume the presence of one or multiple malicious Tor relays, either being compromised by the attacker or originally run by the attacker. Most of the attacks in the literature targeting the de-anonymization of Tor's hidden services falls under this category [24] [25] [26]. This assumption falls within the scope of Tor's threat model stating that Tor provides anonymity against an adversary who can control part of the network.
Flow watermarking attacks are a different category of attacks that also use this assumption. Flow Watermarking attacks are active attacks that depend on the presence of an embedder to inject the watermark in the flow, and the detector(s) that tries to confirm if the watermark is found in the captured flow or not [27] [28]. Flow watermarking attacks are used in the context of Tor anonymity networks to reveal the IP addresses of its hidden services. In their *INFLOW*, Iacovazzi, *et al* [29], they introduced an inverse flow watermark to de-anonymize Tor's hidden services. INFLOW assumes an adversary that controls at least one of the entry guards of the hidden service. The adversary poses as a hidden service client and tries to access the service using its onion address and build a circuit to it. The watermark embedding module is implemented on the corrupted client, while the detection of the watermark happens on the corrupted entry guard run by the adversary. The watermark used depends on introducing bursts of silent traffic in the flow from the client to service. The results obtained in their evaluation reached 96% true positive, however, the detection module can be affected by large packet loss. The watermark can be detected by an observing third-party, which might raise suspicious. Another watermarking attack was introduced by Iacovazzi, *et al* [30] called *DUSTER*. In this attack, the adversary also takes control of multiple Tor nodes. First the adversary scan the dark web for a list of services, and then using a malicious Tor client injects a watermark in the stream contacting one service at a time. The adversary controls several Tor guards acting as watermark detectors. Once a detector spots the watermark in the stream passing through it, it informs the client with the real IP of the service endpoint, and remove the watermark from the stream. The client associate the IP with the service and moves to another service and repeat these steps. In this attack, the adversary is assumed to be able to control enough Tor guards to cover as many rendezvous circuits as required, and that the probability of getting selected as an onion service entry guard is really high. Although the assumption falls within the threat model of Tor, it is still impractical. In general, this attack takes advantage if the congestion control mechanism of Tor and the results showed good accuracy in watermark detection. However, the increase of the detection size, more than 50% of the watermark size, led to decreased detection accuracy.
On the other hand, some attacks aiming to de-anonymize the client can also follow the assumption of compromised Tor nodes. Murdoch, *et al* [7], proposed an active traffic analysis attack based on traffic timing information. The goal of the attack is to identify the communication path from the client to the server. Murdoch's attack follows the threat model of Tor, the adversary in the proposed attack controls a malicious server that actively modulates the delay of the traffic being sent to the client, as well as a compromised Tor relay. The corrupted Tor relay keeps checking other relays to confirm the existence of the delay pattern injected by the server in order to identify the circuit path used by the client. The weak point of this attack is that it requires probing each relay in the Tor network, and with heavy traffic, the accuracy of the correlation function decreases.

V. Analysis

From the aforementioned discussion, we noticed that certain types of attacks are most associated with specific threat models and are best suited for their assumptions. Traffic correlation attacks, for instance, to acquire the flow at multiple points of the network and apply the correlation function to confirm the match. Many of the introduced traffic correlation attacks in the literature choose to match both entry and exit flow of the Tor network, consequently the threat model they use is

Attack	Threat Model	Attack Method	Attack Type	Drawbacks
DeepCor - Nasr, *et al* [13]	Global Adversary	Traffic Correlation	Passive	Correlation accuracy decreases with time
Chakravarty, *et al* [15]	Global Adversary	Traffic Correlation	Active	Congestion in live Tor affects the correlation accuracy
Song, *et al* [14]	Global Adversary	Traffic Correlation	Passive	Offline evaluation with limited dataset
Gilad and Herzberg [20]	Capture Entry Flow	Side-channel	Active	The evaluation on Tor network is not sufficient
Torben - Arp, *et al* [21]	Capture Entry Flow	Side-channel	Active	injected marker not well hidden- limited experiment
Lashkari, *et al* [22]	Capturing Entry Flow	Traffic Classification	Passive	controlled environment was used
He, *et al* [23]	Capture Entry Flow	Traffic Classification	Passive	Less accurate compared to Lashkari's
INFLOW - Iacovazzi, *et al* [29]	Compromising Tor nodes	Flow watermarking	Active	high packet loss ration affects the accuracy
DUSTER - Iacovazzi, *et al* [29]	Compromising Tor nodes	Flow watermarking	Active	Accuracy decreases with the increase of detection size
Murdoch, *et al* [7]	Compromising Tor nodes	Traffic correlation	Active	Heavy traffic affects the accuracy

Table I: Traffic Analysis Attacks Comparison

usually the *global adversary* model. Side-channel attacks, on the other hand, de-anonymize the clients on Tor's network by manipulating some side-channel information in a certain pattern and then attempting to spot that pattern in the traffic flow from the client to the entry guard of Tor's network. The only traffic flow this type of attacks need to capture is the *entry flow*. Moreover, the attacks attempting to identify and separate Tor's traffic from normal traffic necessitates observing the *entry flow* and the accuracy of each attack depends solely on the classification method and the features selected. the category of flow watermarking attacks is following the typical threat model of Tor anonymity network. In these attacks the adversary is assumed to be able to control multiple Tor nodes in order to be able to launch his attack.

VI. Conclusion and Future Work

In this paper, we presented an analysis of a category of attacks that have proven to be effective on the anonymity of Tor's network. The threat model adopted by each attack described can be directly related to its practicality. For the discussed attacks in this survey, launching the attack successfully in the real networks requires a solid evaluation of the probability of maintaining the compromised entities for the needed duration. Our future plan includes studying other types of traffic analysis attacks, moreover we plan to conduct a comprehensive evaluation of the probability of successfully launching different types of traffic analysis attacks against the Tor network.

VII. Acknowledgement

We would like to acknowledge Qatar University graduate assistantship in KINDI center for computing research. This work was also made possible by NPRP grant 7-1469-1-273 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

[1] R. Dingledine, N. Mathewson, S. Murdoch, and P. Syverson, "Tor: The Second-Generation Onion Router (2014 DRAFT v1)," *Cl.Cam.Ac.Uk*, 2014.

[2] "Configuring Hidden Services for Tor."

[3] A. Back, M. Ulf, and A. Stiglic, "Traffic Analysis Attacks and Trade-Offs in Anonymity Providing Systems," *International Workshop on Information Hiding*, pp. 245–257, 2001.

[4] A. Johnson, R. Jansen, M. Sherr, P. Syverson, and W. Dc, "Users Get Routed : Traffic Correlation on Tor by Realistic Adversaries," *CCS '13 Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pp. 337–348, 2013.

[5] M. Alsabah and I. Goldberg, "Performance and Security Improvements for Tor : A Survey," *ACM Computing Surveys (CSUR)*, vol. 49, no. 2, pp. 1–41, 2014.

[6] "Tor: Onion Service Protocol."

[7] S. J. Murdoch, G. Danezis, S. Murdoch, and G. Danezis, "Low-Cost Traffic Analysis of Tor,"

[8] P. Syverson and M. Reed, "Towards an Analysis of Onion Routing Security," *Designing Privacy Enhancing Technologies*, pp. 96–114, 2001.

[9] M. Perry, "Experimental Defense for Website Traffic Fingerprinting," 2011.

[10] R. Johnson, "Touching from a Distance : Website Fingerprinting Attacks and Defenses," *CCS '12 Proceedings of the 2012 ACM conference on Computer and communications security*, pp. 605–616, 2012.

[11] K. M. Abdullah Qasem, Sami Zhioua, "Finding a Needle in a Haystack: The Traffic Analysis Version," *Proceedings on Privacy Enhancing Technologies*, pp. 270–290, 2019.

[12] K. Kohls, "DigesTor : Comparing Passive Traffic Analysis Attacks on Tor," *European Symposium on Research in Computer Security*, vol. 1, 2018.

[13] M. Nasr, A. Bahramali, and A. Houmansadr, "DeepCorr : Strong Flow Correlation A acks on Tor Using Deep Learning," *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1962–1976, 2018.

[14] M. Song, G. Xiong, Z. Li, J. Peng, and L. Guo, "A De-anonymize attack method based on traffic analysis," *2013 8th International Conference on Communications and Networking in China (CHINACOM)*, pp. 455–460, 2013.

[15] S. Chakravarty, M. V. Barbera, G. Portokalidis, M. Polychronakis, and A. D. Keromytis, "On the Effectiveness of Traffic Analysis Against Anonymity Networks Using Flow Records," *International Conference on Passive and Active Network Measurement*, pp. 1–10, 2014.

[16] J. A. Stone, N. Saxena, and H. Dogan, "Systematic Analysis : Resistance to Traffic Analysis Attacks in Tor System for Critical Infrastructures," *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2832–2837, 2018.

[17] M. Amar and I. Mohd, "A Survey on Tor Encrypted Traffic Monitoring," vol. 9, no. 8, 2018.

[18] Y. Sun, A. Edmundson, L. Vanbever, E. T. H. Zürich, O. Li, J. Rexford, M. Chiang, P. Mittal, A. Edmundson, L. Vanbever, and J. Rexford, "RAPTOR : Routing Attacks on Privacy in Tor," *Proceedings of the 24th USENIX Security Symposium*, 2015.

[19] M. Nasr, A. Houmansadr, and A. Mazumdar, "Compressive Traffic Analysis : A New Paradigm for Scalable Traffic Analysis," *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017.

[20] Y. Gilad and A. Herzberg, "Spying in the Dark : TCP and Tor Traffic Analysis," *International Symposium on Privacy Enhancing Technologies, Springer*, pp. 100–119, 2012.

[21] D. Arp, F. Yamaguchi, and K. Rieck, "Torben : A Practical Side-Channel Attack for Deanonymizing Tor Communication Categories and Subject Descriptors," *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, pp. 597–602, 2015.

[22] A. H. Lashkari, G. D. Gil, M. Saiful, I. Mamun, and A. A. Ghorbani, "Characterization of Tor Traffic using Time based Features," *3rd International Conference on Information Systems Security and Privacy*, no. Cic, pp. 253–262, 2017.

[23] G. He, "Inferring Application Type Information from Tor Encrypted Traffic," *2014 Second International Conference on Advanced Cloud and Big Data*, pp. 220–227, 2014.

[24] R. Jansen and A. Johnson, "The Sniper Attack : Anonymously Deanonymizing and Disabling the Tor Network," *NDSS Symposium 2014*, no. February, pp. 23–26, 2014.

[25] A. Kwon, M. Alsabah, D. Lazar, M. Dacier, and S. Devadas, "Circuit Fingerprinting Attacks : Passive Deanonymization of Tor Hidden Services," *USENIX Security*, pp. 287–302, 2015.

[26] A. Biryukov, I. Pustogarov, and R.-p. Weinmann, "Trawling for Tor Hidden Services : Detection , Measurement , Deanonymization," *2013 IEEE Symposium on Security and Privacy*, pp. 80–94, 2013.

[27] A. Iacovazzi and Y. Elovici, "Network Flow Watermarking : A Survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 512–530, 2017.

[28] A. Houmansadr, N. Kiyavash, and N. Borisov, "RAINBOW : A Robust And Invisible Non-Blind Watermark for Network Flows," *NDSS Symposium 2009*.

[29] A. Iacovazzi, S. Sarda, and Y. Elovici, "I NFLOW : Inverse Network Flow Watermarking for Detecting Hidden Servers," *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pp. 747–755, 2018.

[30] A. Iacovazzi, D. Frassinelli, and Y. Elovici, "The D USTER Attack : Tor Onion Service Attribution Based on Flow Watermarking with Track Hiding," *22nd International Symposium on Research in Attacks, Intrusions and Defenses*, pp. 213–225, 2019.