Poster: Fingerprinting Hidden Service Circuits from a Tor Middle Relay

Marc Juarez
imec-COSIC KU Leuven
marc.juarez@kuleuven.be

Rob Jansen
U.S. Naval Research Laboratory
rob.g.jansen@nrl.navy.mil

Rafael Galvez
imec-COSIC KU Leuven
rafael.galvezvizcaino@kuleuven.be

Tariq Elahi
imec-COSIC KU Leuven
tariq.elahi@esat.kuleuven.be

Claudia Diaz
imec-COSIC KU Leuven
claudia.diaz@esat.kuleuven.be

Matthew Wright
Rochester Institute of Technology
matthew.wright@rit.edu

***Abstract*—Kwon et al. recently showed that *circuit fingerprinting* attacks could be used to identify hidden service circuits, which is a key step towards linking Tor users and their activity online. In this paper, we explore an improvement to their attack that uses random forests, which achieves similar accuracy while being more robust to simple countermeasures against it. Additionally, we perform our attack from a *middle* node, for which an attacker needs less resources and can leverage guard fingerprinting to deanonymize users. Our evaluation shows the attack can be effectively deployed at the middle with 99.98% accuracy.**

I. Introduction

Anonymity systems like Tor provide online privacy for millions of users every day. Tor users connect to websites and other services through *circuits*, paths of relays that anonymize the connections between the users and their destinations. Tor also offers *hidden services* (HSes), a way for servers (typically web servers) to be accessed only through Tor such that their location is hidden from the clients.

Unfortunately, Kwon et al. recently showed that the Tor circuits that connect to HSes can be distinguished from other Tor circuits, in an attack called *circuit fingerprinting* [5]. They then show that knowing a circuit is being used for a hidden service makes it particularly vulnerable to *website fingerprinting*, an attack that can be used to identify which website a user is visiting. While website fingerprinting can be difficult in realistic settings, such as when users could be visiting any of the sites on the entire Web [4], the relatively small number of hidden services makes the attack more feasible.

In their work, Kwon et al. use a local adversary model, one who can either eavesdrop on the user's connections or who controls the first relay on the user's circuit. While this model is surely a realistic one for some attackers, other attackers may not have that level of access. In particular, an attacker who wants to control the first relay (also known as the *guard*) on a particular user's circuit will need to satisfy the necessary requirements to obtain the *guard flag*, an indicator that the relay is suitable for regular use as the first hop. Since a user continuously uses a single guard for up to nine months [2], the attacker then must maintain those relays for a long period of time just to have a chance to be selected as the next guard.

A less powerful attacker model is one who runs only middle nodes that sit between the guard and the last relay on the user's circuits. This relay offers the least visibility to the attacker, but it is the easiest position to get. In particular, the middle relay is essentially selected from among all the possible middle relays based on its relative proportion of bandwidth. An adversary who controls a set of low-bandwidth, low-reliability nodes cannot get many guards, but could act as the middle node for many of the user's circuits over time. The middle node cannot identify the client, but it can see the guard node. As others have pointed out [7], [6], [3], [8], knowing the guard node allows the attacker to create a weak pseudonymous identifier for the client and link her activity over multiple browsing sessions.

In this paper, we present an improvement on the original circuit fingerprinting attack that uses random forests. Furthermore, we show the attack can be applied effectively on Tor middle nodes, partially deanonymizing users from the middle node position. These results further motivate the need for effective defenses against traffic fingerprinting attacks.

II. Data Collection

We have automated our data collection using `tor-browser-crawler`[1], a web crawler that allows us to visit a list of HS URLs with the Tor Browser. As the crawler visits URLs, we use `tshark` to capture the network traffic generated by visiting web pages. We based our collection methodology on previous studies on website fingerprinting [10].

The list of URLs that we are using has been extracted from Ahmia[2], the most popular search engine for onion pages. Ahmia exposes a REST API that allows us to query a ranked list of onion URLs by click popularity. Ahmia periodically checks onion services and collects statistics about their uptime. We have used those statistics to obtain a list of the 1,000 most popular and stable onion services. Before starting the crawls, we have used `torsocks` and `curl` to remove from the list onion sites that are down. We rely on Ahmia's blacklist to avoid downloading illegal HS pages.

[1] https://github.com/webfp/tor-browser-crawler
[2] https://ahmia.fi

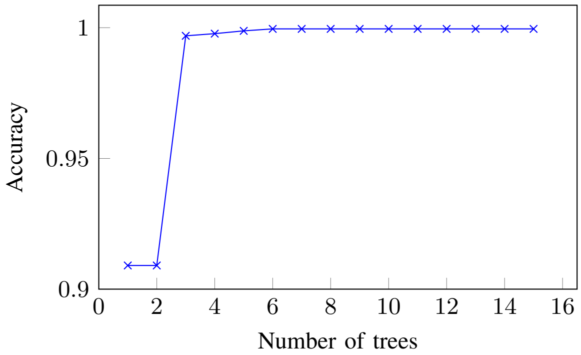

Figure 1: 10-fold cross-validation with three repetitions and grid-search to find the number of trees (x-axis) that maximizes the random forest accuracy (y-axis). Note the y-axis does not start at zero.

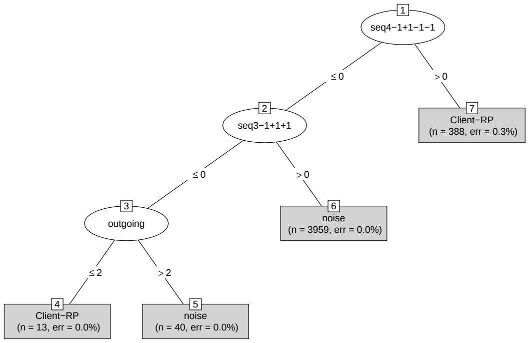

Figure 2: Decision tree built from reproducing the original circuit fingerprinting attack modified to distinguish only between client-to-Rendezvous-Point and regular circuits. $seq3$ and $seq4$ are the three and four first cells exchanged during circuit creation and are the most distinguishing features, followed by the number of outgoing cells.

We ran a Tor relay with a capacity of 1 MBps for a week and used a modified Tor client to pin the middle node of all the client's circuits. The middle node logged Tor cell-level information that was used to train the classifiers. To avoid the risks to regular Tor users connecting to our middle, we only collect this information on circuits originating at our own Tor client. For that we used a signaling mechanism that flags circuits started by our crawler to the middle, so that it only logs cells flowing through that circuit.

III. Evaluation

Kwon et al. used a decision tree classifier to distinguish between different types of Tor circuits. In this paper, we first reproduced their results for the simpler problem of telling client-to-Rendezvous-Point circuits apart from the rest of circuits. We obtained a perfect classification score, namely 100% accuracy, on their dataset (depicted in Figure 2).

Decision trees are simple and easy-to-analyze models but are not robust to small modifications of the features they are based on. Since the circuit fingerprinting attack relies on analyzing the initial sequence of cells, simple countermeasures that added a few dummy cells during the circuit construction would defeat the attack as presented in the original paper.

A straight-forward approach to obtain a statistical model from a decision tree is to ensemble several trees, by randomizing the feature and training data selection of each individual tree. These models are known as *random forests* and tend to generalize better than simple decision trees, thus becoming more robust against basic defense strategies.

We evaluated the random forest attack on both, traffic traces collected at the client and at the middle nodes. Figure 1 shows the accuracy of our random forest on the traffic traces collected at the middle. Despite being at the middle we observe an accuracy comparable to the one at the client. This result implies that an adversary with low resources could effectively deploy a more robust version of the circuit fingerprinting attack from Tor middles.

IV. Conclusion and Future Work

This paper lays the ground for future research on defenses that protect against traffic fingerprinting attacks at any position in a Tor circuit. As future work, we plan to investigate countermeasures against the circuit fingerprinting attack and we will use the random forest version presented in this paper to evaluate them. In addition, we have shown that the circuit fingerprinting attack poses a threat if deployed at the middle position and point out that future defenses should take this into account.

References

[1] G. Cherubin, J. Hayes, and M. Juarez, "Website Fingerprinting Defenses at the Application Layer," in *Privacy Enhancing Technologies Symposium (PETS)*. De Gruyter, 2017, pp. 168–185.

[2] R. Dingledine and G. Kadianakis, "One fast guard for life (or 9 months."

[3] N. Hopper, E. Y. Vasserman, and E. Chan-Tin, "How much anonymity does network latency leak?" *ACM Transactions on Information and System Security*, vol. 13, no. 2, February 2010.

[4] M. Juarez, S. Afroz, G. Acar, C. Diaz, and R. Greenstadt, "A critical evaluation of website fingerprinting attacks," in *ACM Conference on Computer and Communications Security (CCS)*. ACM, 2014, pp. 263–274.

[5] A. Kwon, M. AlSabah, D. Lazar, M. Dacier, and S. Devadas, "Circuit fingerprinting attacks: passive deanonymization of tor hidden services," in *USENIX Security Symposium*. USENIX Association, 2015, pp. 287–302.

[6] P. Mittal, A. Khurshid, J. Juen, M. Caesar, and N. Borisov, "Stealthy traffic analysis of low-latency anonymous communication using throughput fingerprinting," in *Proceedings of the 18th ACM conference on Computer and Communications Security (CCS 2011)*, October 2011.

[7] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of Tor," in *Proceedings of the 2005 IEEE Symposium on Security and Privacy*. IEEE CS, May 2005.

[8] L. Overlier and P. Syverson, "Locating hidden servers," in *IEEE S&P*, 2006.

[9] A. Panchenko, F. Lanze, A. Zinnen, M. Henze, J. Pennekamp, K. Wehrle, and T. Engel, "Website fingerprinting at internet scale," in *Network & Distributed System Security Symposium (NDSS)*. IEEE Computer Society, 2016, pp. 1–15.

[10] T. Wang and I. Goldberg, "Improved Website Fingerprinting on Tor," in *ACM Workshop on Privacy in the Electronic Society (WPES)*. ACM, 2013, pp. 201–212.