Multipathing Traffic to Reduce Entry Node Exposure in Onion Routing

Jan Pennekamp*, Jens Hiller*, Sebastian Reuter*, Wladimir De la Cadena†,
Asya Mitseva†, Martin Henze§, Thomas Engel†, Klaus Wehrle*, Andriy Panchenko‡
**Communication and Distributed Systems*, *RWTH Aachen University*, Germany · {lastname}@comsys.rwth-aachen.de
†*University of Luxembourg*, Luxembourg · {wladimir.delacadena, asya.mitseva, thomas.engel}@uni.lu
§*Cyber Analysis & Defense*, *Fraunhofer FKIE, Bonn-Bad Godesberg*, Germany · martin.henze@fkie.fraunhofer.de
‡*IT Security*, *Brandenburg University of Technology, Cottbus*, Germany · andriy.panchenko@b-tu.de

***Abstract*—Users of an onion routing network, such as Tor, depend on its anonymity properties. However, especially malicious entry nodes, which know the client's identity, can also observe the whole communication on their link to the client and, thus, conduct several de-anonymization attacks. To limit this exposure and to impede corresponding attacks, we propose to multipath traffic between the client and the middle node to reduce the information an attacker can obtain at a single vantage point. To facilitate the deployment, only clients and selected middle nodes need to implement our approach, which works transparently for the remaining legacy nodes. Furthermore, we let clients control the splitting strategy to prevent any external manipulation.**

I. Introduction

Onion routing promises clients to protect their anonymity against local adversaries, such as malicious entry onion routers (ORs) or network operators (ISPs), by preventing a linking of sender and receiver. Hence, it enables users to anonymously access web content or exchange sensitive data. Some applications are censorship bypassing, fostering of free speech, whistleblower protection, or reduced tracking for the Internet or IoT [1]. Consequentially, adversaries have an incentive to attack such networks to regain insight into the users' behavior.

Attacks on Tor already reveal that adversaries try to deanonymize participants or to link communication partners [2]. Local adversaries that can observe the onion routing users have an advantageous position to mount such attacks as they are aware of the client's identity and have access to the traffic on the *first OR-hop*, i.e., between the client and the onion network. Especially when striving to deploy an entry node, potentially malicious operators face almost no challenges. To defend users against local attackers, i.e., malicious entries or on-path adversaries (e.g., the client's ISPs), we propose to distribute the traffic over multiple entry nodes (called guard nodes in Tor) over possibly unrelated network connections (e.g., utilizing different ISPs via DSL, Wi-Fi, satellite and/or cellular networks) to finally merge it at the middle OR. We motivate this design by early results that indicate better security when attackers control only a share of the traffic [2].

II. Target Scenario

Our multipathing limits the exposure to a (single) entry node and the risks coupled with only a single connection into the onion routing network. We specifically target to protect clients against traffic analysis and website fingerprinting (WFP) attacks [3], [4] by local adversaries. These attacks commonly make use of a single vantage point between the client and the onion routing network, which allows them to identify the user and—using machine-learning based traffic pattern matching—the accessed websites. To counter WFP, we propose to split traffic over multiple entry nodes instead of a—possibly malicious [5]—single entry node. Thereby, we must defeat DoS attacks that (i) can reduce the multipathing back to a single path only, or (ii) introduce new vulnerabilities to the modified network (cf. the past sniper attack [6]). Besides protecting against WFP, we also aim to complicate timing attacks (cf. raptor attack [7]) in multipathing scenarios where the attacker has no full control over all user-chosen entries.

In the past, research already applied multipathing to onion routing [8] to mitigate the threats introduced by using a single route through the network. In contrast to them, our approach does not require changes to the complete onion routing network, or scarce and often already overloaded exit ORs. Specifically, we target the following design goals: **(G1)** limiting the changes to only a few nodes and without modifying scarce entry or exit ORs, **(G2)** maintaining compatibility with today's onion routing, **(G3)** avoiding a negative impact on the performance, and **(G4)** handing all multipathing control to the user to exclude the influence of possibly malicious ORs.

III. Design of Multipathing for Onion Routing

We illustrate our envisioned approach in Figure 1. It allows users to multipath traffic over independent circuits between the client and the middle OR, i.e., those nodes serve as splitting and reassembling points. To this end, we introduce a buffer mechanism to deal with cells that arrive out of order due to varying network latencies and available bandwidth of the different circuits. Otherwise, further processing would fail as the counter-mode encryption requires cells to arrive in order. Our traffic distribution over multiple entries should protect users against attacks of a malicious entry. Moreover, the design is also effective against attacks by an ISP if the (sub)circuits are spawned over network connections of different network operators (e.g., utilizing different Wi-Fi access points). We now discuss how we envision to achieve our design goals.

DOI: 10.1109/ICNP.2019.8888029

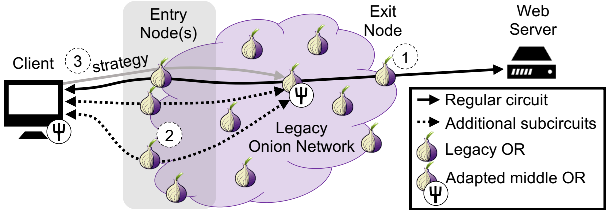

Fig. 1. To **establish a multipath onion routing circuit** with a web server, (1) the client first establishes a regular onion circuit over an adapted middle OR. Then, (2) it reduces his exposure to each entry by building subcircuits to the middle OR, effectively multipathing his connection. To enable the middle to match all (sub)circuits, the client transmits a cookie on each of them. Finally, (3) the clients provides the middle OR with the splitting strategy. Our design limits changes to only some middle ORs and the client.

Transparent Local Changes (G1 & G2). Our design is transparent for the unmodified onion network, i.e., the remaining network operates without any changes. For Tor, we need to deploy changes only to the client and some middle ORs. Adapting the middle ORs is reasonable as the number of middle ORs is high and Tor does not pose special requirements on them. All modified middle ORs continue to support regular connections without enabled traffic splitting.

Splitting Control (G4). The client determines how to split the traffic over the circuits and then shares this strategy with the middle OR (cf. Figure 1 (3)). Hence, only the client is in charge of the traffic distribution, which plays a key role in thwarting attacks. While a malicious middle OR could still deviate from the received strategy, clients can detect this event and react. For the splitting, we can rely on different strategies, e.g., round robin or random [8]. We plan a security analysis detailing the effectiveness of these strategies in future work.

Unchanged Performance (G3). We target to realize multipathing without user-noticeable overhead for (multiple) circuit establishments as circuits are typically built in advance. To meet strict time constraints in certain scenarios, a client could initially transfer cells over the original circuit without traffic splitting, first enabling splitting once all subcircuits are set up. Real-world circuit setup overheads for guard and middle ORs will be part of a future evaluation. During operation, we expect the overhead to be negligible as we only introduce a few managements cells (between client and middle OR) to control the traffic splitting. Related work even achieved performance improvements with end-to-end multipathing [9].

IV. Security Discussion

Our approach requires middle ORs to buffer traffic for in-order reassembly. As such buffering is already needed in standard Tor, we can adapt already existing countermeasures against buffer reservation attacks [6]. From the viewpoint of clients, middle ORs can match subcircuits of a client but not obtain its identity as no direct client-to-middle connection exists. Finally, the client determines both the number of subcircuits and the splitting strategy (cf. G4). Consequentially, it can easily detect deviations introduced by malicious ORs.

V. Related Work & Evaluation

For an extensive analysis of splitting approaches in related work, we refer to a recent survey [8]. In contrast to existing approaches, e.g., Conflux [9], which mainly target an improved performance by multipathing traffic between entry and exit ORs, our motivation for splitting traffic over multiple entries is to provide a countermeasure against timing and traffic analysis attacks without modifying already overloaded exit ORs while still removing a single path to an entry OR.

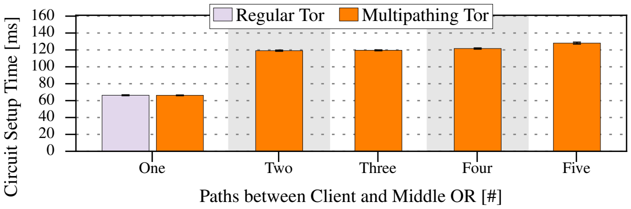

Fig. 2. Our multipathing support adds no noticeable overhead for traditional Tor circuit establishment. While additional paths introduce overhead by design, the client can still establish multiple subcircuits in parallel.

We conducted our evaluation over 400 runs each in a local testbed (8x Intel i5@3.3 GHz, 16 GB RAM) to measure the design overhead. Adding subcircuits for multipathing doubles the setup runtime as subcircuits are established subsequently (cf. Figure 2). However, multiple subcircuits can be established in parallel. For low latency, payload can already be transferred with a single path while sacrificing our security properties.

Overall, our multipathing introduces acceptable overheads.

VI. Conclusion & Future Work

We propose to split onion routing traffic over multiple entry nodes and reassemble traffic at the middle OR to tackle attacks that root in the usage of a connection between the client and a single entry. Our presented design aligns well with the design of today's onion routing networks as changes are only required locally for a subset of existing nodes. For future work, we plan to evaluate the performance as well as to analyze traffic splitting strategies in view of their effectiveness against the attacks. Especially w.r.t. WFP attacks, a combination with an integrated padding approach might reduce the required overhead, i.e., performance loss, when compared to a (successful) padding scheme over a single circuit.

Acknowledgments: Partially funded by the Luxembourg National Research Fund (FNR) within the CORE Junior Track project PETIT.

References

[1] J. Hiller *et al.*, "Tailoring Onion Routing to the Internet of Things: Security and Privacy in Untrusted Environments," in *IEEE ICNP*, 2019.
[2] M. AlSabah and I. Goldberg, "Performance and Security Improvements for Tor: A Survey," *ACM CSUR*, vol. 49, no. 2, 2016.
[3] A. Panchenko *et al.*, "Website Fingerprinting at Internet Scale," in *NDSS*, 2016.
[4] P. Sirinam *et al.*, "Deep Fingerprinting: Undermining Website Fingerprinting Defenses with Deep Learning," in *ACM CCS*, 2018.
[5] Q. Li *et al.*, "A Stealthy Attack Against Tor Guard Selection," *International Journal of Security and Its Applications*, vol. 9, no. 11, 2015.
[6] R. Jansen *et al.*, "The Sniper Attack: Anonymously Deanonymizing and Disabling the Tor Network," in *NDSS*, 2014.
[7] Y. Sun *et al.*, "RAPTOR: Routing Attacks on Privacy in Tor," in *USENIX Security*, 2015.
[8] W. De la Cadena *et al.*, "Analysis of Multi-path Onion Routing-Based Anonymization Networks," in *IFIP WG 11.3 DBSec*, 2019.
[9] M. AlSabah *et al.*, "The Path Less Travelled: Overcoming Tor's Bottlenecks with Traffic Splitting," in *PETS*, 2013.