Donar: Anonymous VoIP over Tor

Yérom-David Bromberg, Quentin Dufour, Davide Frey, Etienne Rivière

Donar: Anonymous VoIP over Tor

Yérom-David Bromberg, Quentin Dufour, Davide Frey
Univ. Rennes - Inria - CNRS - IRISA, France

Etienne Rivière
UCLouvain, Belgium

Abstract

We present DONAR, a system enabling anonymous VoIP with good quality-of-experience (QoE) over Tor. No individual Tor link can match VoIP networking requirements. DONAR bridges this gap by spreading VoIP traffic over *several* links. It combines active performance monitoring, dynamic link selection, adaptive traffic scheduling, and redundancy at no extra bandwidth cost. DONAR enables high QoE: latency remains under 360 ms for 99% of VoIP packets during most (86%) 5-minute and 90-minute calls.

1 Introduction

Tor [20] is by far the largest anonymization network with over 6,000 relay nodes distributed worldwide. Tor has been very successful for applications such as web browsing with, e.g., TorBrowser, but is generally considered inadequate for latency-sensitive applications [31,66]. Voice-over-IP (VoIP) is one such application that has become the *de facto* solution for global voice calls. Being able to deploy VoIP over Tor would immediately benefit privacy-conscious users by enabling simple, efficient, and safe voice communication answering two objectives: (i) protecting the content of the communication from adversaries, i.e., using end-to-end encryption, and (ii) hiding metadata and in particular the identity of communicating partners. Metadata may, indeed, be used to infer private information, e.g., uncovering a journalist's sources [27] or illegally gathering information about employees [60].

Providing good-quality interaction between VoIP users, i.e., a good Quality-of-Experience (QoE), requires good network Quality-of-Service (QoS) and in particular low and stable latency [28, 35, 68], as we detail in **Section 2**. This comes in tension with the way Tor is designed [31,66]: Tor links[1] implement multi-hop communication for TCP traffic using *onion routing* over pre-established circuits formed of several relays, which leads to high and unstable latencies. Surprisingly, Sharma *et al.* [70] recently posited that using Tor *as is* would be sufficient to obtain the stable and low latencies required by high-QoE VoIP. This statement is, unfortunately, incorrectly grounded. Three biases in their analysis led to this conclusion: (1) they only consider average latencies, while VoIP QoE is primarily determined by tail latency (99th percentile with a standard codec) [57], (2) they measure performance for only 30 seconds, a much shorter duration than an average call [33], and (3) they only consider the case of one-way anonymity, i.e., when the callee is not anonymous. We present in **Section 3** our analysis of Tor links' performance considering these elements and conclude that the use of a Tor link *as is* does not, in fact, allow VoIP with sufficient QoE.

TorFone [24] attempts to overcome Tor latency issues by duplicating VoIP traffic over two statically chosen links. However, even if going in the right direction, TorFone's strategy turns out to be ineffective due to the large variability of link performance over time, as we demonstrate in **Section 6**.

Alternative anonymization networks targeting voice communication were also proposed recently, e.g., Herd [50] and Yodel [49]. These systems, however, are yet to be deployed and need to reach a sufficient scale to be efficient, i.e., to provide sufficient bandwidth for a large number of geographically-distributed users.

Motivations. We are interested in providing VoIP support over a *readily-available* anonymization network. More specifically, we target a deployment using (1) legacy VoIP applications and (2) the existing, unmodified Tor network. We do not wish to propose design changes to Tor, or a novel anonymity network [49, 50, 64, 73, 76], and neither do we want to overcome Tor's existing security flaws. We believe that these lines of work are, in fact, orthogonal to our own.

While our observation of the performance of Tor (presented in **Section 3**) confirms that a *single* Tor link cannot provide the stable and low latency required by high-QoE VoIP, it also allows us to make a case for dispatching traffic over *multiple* links. Unlike TorFone, our strategy multiplexes traffic over a *dynamically selected* set of Tor links using a smart scheduling mechanism. Our motivation is that the use of multiple dynamically and adequately chosen links, together with controlled

[1] We use in this paper the generic term *link* to denote the unidirectional TCP channel that is exposed to applications by the Tor client.

and smart content redundancy, can mask the transient faults and latency spikes experienced by individual links.

Contributions. We present the design and implementation of DONAR, a user-side proxy interfacing a legacy VoIP application to the existing Tor network (**Section 4**).

DONAR enforces *diversity* in the paths used for transmitting VoIP packets, i.e., using distinct Tor links. In addition, it leverages *redundancy* by sending the same VoIP packet several times using different links. This redundancy does not, in fact, add additional bandwidth costs for the Tor network beyond those incurred by the setup and maintenance of these multiple links. We leverage, indeed, the fact that Tor only transmits 514-Byte cells over the network to protect users against traffic analysis [53, 59]. DONAR takes advantage of the available padding space to re-transmit previous VoIP packets. Diversity and redundancy mask the impact of the head-of-line blocking implied by the TCP semantics of Tor links, whereby an entire stream of packets may get delayed by a single belated one.

DONAR builds on the following key technical components:

- The *piggybacking* of VoIP packets in the padding space of Tor cells enables redundancy without incurring additional bandwidth costs on the Tor network.
- A *link monitoring* mechanism observes and selects appropriate links, switching rapidly between them when detecting performance degradation.
- Two *scheduling* strategies for selecting links when transmitting VoIP packets enable different tradeoffs between cost and robustness.

We further analyze in **Section 5** how attacks on Tor can affect the security properties of DONAR. In particular, we discuss how different DONAR configurations implement different tradeoffs between Quality-of-Experience and security.

We evaluate DONAR over the Tor network and present our findings in **Section 6**. We use VoIP-traffic emulation as well as the off-the-shelf `gstreamer` [26] VoIP client using the OPUS [14] audio codec. We assess the performance of DONAR against the VoIP requirements detailed in Section 2 and compare it with the approach followed by TorFone [24]. Our results show that DONAR, using alternatively 6 out of 12 carefully monitored and dynamically selected onion links, achieves high QoE with latency under 360 ms and less than 1% of VoIP frame loss for the entire duration of a large number (86%+) of 5-minute and 90-minute calls, with no bandwidth overhead for its optimized configuration (i.e., alternate sending over different links) and an overhead similar to that of TorFone for its default configuration.

We detail related work and conclude in **Sections 7 and 8**.

2 VoIP networking requirements

DONAR aims at Providing good Quality-of-Experience (QoE) for anonymous VoIP while limiting the costs imposed on the Tor infrastructure. We base our analysis of QoE requirements on recommendations by the International Telecommunication Union (ITU) [35–37]. The ITU defines good QoE as the combination of the following guarantees: (1) uninterrupted calls, (2) good voice quality, and (3) support for interactive conversations. We analyze these requirements and derive our network QoS objectives, summarized in Table 1.

Metric	**Objective**
Dropped calls rate	$\leq$ 2% for 90-minute calls
Packet loss rate	$\leq$ 1%
Bandwidth	$\geq$ 32 kbps (4.3 kB/s)
One way delay (99th perc. *ideal*)	$\leq$ 150 ms - T_{frame} - T_{buffer}
One way delay (99th perc. *max*)	$\leq$ 400 ms - T_{frame} - T_{buffer}

Table 1: VoIP network performance requirements, following the recommendations of the International Telecommunication Union [35] and applying them to the OPUS codec [74, 75].

VoIP protocols. VoIP requires two types of protocols. A signaling protocol such as the Session Initiation Protocol (SIP) [67] makes it possible to locate a correspondent and negotiate parameters for the communication. The signaling protocol only impacts QoE with delays upon the establishment of the call. When the call is established, a protocol such as the UDP-based Real-time Transport Protocol (RTP) [69] is used to transmit VoIP audio frames encoded using a codec, whose configuration is negotiated by the signaling protocol. QoE is primarily impacted by this codec and its ability to deal with hazards in network QoS, as we detail next.

Impact and choice of the audio codec. Bandwidth, latency, or maximum packet loss requirements depend on the audio codec used by the VoIP application. We base our analysis on the state-of-the-art open audio codec OPUS, which we also use in our evaluations. OPUS is a widely-used, loss-tolerant audio codec developed by the Xiph.Org Foundation and standardized by the IETF [74, 75]. It targets interactive, low-delay communication and computational efficiency. OPUS has been consistently ranked in comparative studies as the highest-quality audio format for low and medium bit rates [32, 41]. We emphasize that our analysis would be similar for other open codecs, e.g., the Internet Low Bit Rate Codec (iLBC) [4] or Xiph.Org Foundation's former codecs Vorbis [7] and Speex [30].

First guarantee: *no call interruption*. A call interruption is the most impacting event on user-perceived QoE. The ITU does not provide a recommendation for general networks but recommends at most 2% dropped calls for VoIP over 4G [37]. We adopt the same goal but need to define a time span on which to evaluate this metric. Holub *et al.* [33] provided us with a dataset of more than 4M call durations (Figure 1). Its analysis confirms that call duration follows a log-normal distribution considered as standard for voice calls. We observe an average call duration slightly above 3 minutes, with 90% of

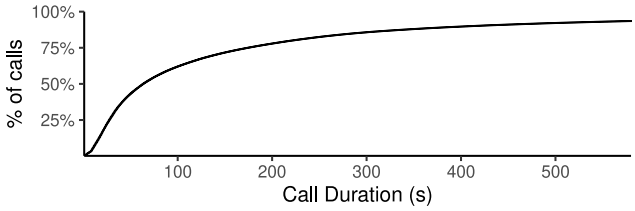

Figure 1: Call Duration ECDF on a 4M calls dataset provided by Holub *et al.* [33] zoomed on the first 10 minutes.

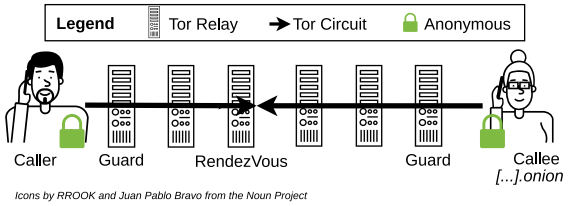

Figure 2: Structure of a Tor link with onion services.

calls lasting less than 10 minutes. However, we still observe 1,040 calls lasting for 90 minutes or more which is characteristic of the long tail of the distribution. As this value matches the limitation set by major representative carriers [34, 77], we evaluate reliability for calls over this duration.

Second guarantee: *good voice quality*. Users want to clearly hear their communication partners. Voice quality depends both on the bitrate being used and the amount of packet loss: (1) Listening tests with OPUS [14, 32] concluded that a bitrate of 32 kbps is sufficient to offer a sound quality that test users cannot distinguish from a reference unencoded version of the recording. We set, therefore, this bitrate as the minimum required bandwidth that we must offer to the VoIP application. (2) OPUS provides two mechanisms to mask the impact of lost packets: a domain-specific one, named Packet Loss Concealment (PLC) and a generic one, via redundancy, named Forward Erasure Coding (FEC)[2] [72]. Han *et al.* [28] studied the perceived quality of a call on various packet rates. This study shows that while PLC compensates for packet loss, the perceived voice quality nonetheless decreases quickly: a 1% packet loss is essentially unnoticed, while 10% packet loss results in usable but degraded call conditions. Based on these results, we set as a requirement a packet loss of at most 1%.

Third guarantee: *interactive conversations*. In addition to an uninterrupted and good-quality voice signal, users of voice calls expect to be able to exchange information interactively, e.g., be able to seamlessly synchronize on when to stop and start talking in a conversation.

Interactivity primarily depends on latency [68]. The ITU published recommendation G.114 [35] on mouth-to-ear latency in voice calls. This recommendation indicates that a delay below 150 ms is unnoticeable for users, compared to a direct voice conversation. We set, therefore, this value as our ideal latency. On the other hand, the recommendation stipulates that delays must remain below 400 ms to make an interactive call possible under good conditions. Higher latencies result in synchronization difficulties and significantly reduce user-perceived QoE. We set this threshold of 400 ms as our maximum acceptable mouth-to-ear latency.

We emphasize that the actual network latency for transmitting VoIP frames is only a subset of mouth-to-ear latency. Additional latency is introduced by (1) audio capture and playing, (2) packetization, and (3) buffering. Once digitized, audio is encapsulated, every T_{frame} ms, into frames that will form a packet. OPUS enables configurable values for T_{frame} from 2.5 to 60 ms.

We consider an ideal jitter-buffer model similar to the one by Moon *et al.* [57]. This model delays all frames by the maximum or nth percentile of the observed latency and allows frame drops. Moon *et al.* [57] and others [44, 52] have proposed jitter-buffer implementations performing close to this theoretical optimum. Therefore, we consider T_{buffer}, the unnecessary delay added by a wrong jitter-buffer configuration as negligible. Finally, as we allow a 1% frame drop, we consider the 99th latency for our mouth-to-ear delay constraints.

3 VoIP over Tor: How bad is it?

Tor [20] is a large-scale network that enables users to access remote resources anonymously. Tor relies on *onion routing*: it relays traffic through *circuits* consisting of at least two relays (three by default) chosen from more than 6,000 dedicated nodes. The first relay in a circuit is known as the *Guard*. The Tor client chooses a small set of n (by default[3], $n = 2$) possible guards. Thereafter, it builds circuits by using one guard from this set, choosing the remaining relays randomly from the list of all available relay nodes.

Tor enables both connections to the regular Internet (referred to as *Exit*) and to other Tor users (referred to as *Onion Services*). In contrast to the *Exit* mode, *Onion Services* provide two-way anonymity by default. The Tor client on the caller's side connects to an anonymous onion service (set up by the callee's Tor client). In doing so, it creates a Tor route, i.e., the concatenation of two Tor circuits, one from the caller to a rendezvous relay, and another from the callee to the same rendezvous relay. In this paper, we use the term *link* to refer to the TCP connection over this route that the Tor client exposes to the application. Figure 2 illustrates a Tor link based on an onion service used for transmitting VoIP frames.

Tor seeks to prevent adversaries from inferring communicating parties. To this end, at least one relay in the route should lie in an administrative domain that the adversary cannot observe. Furthermore, to prevent traffic analysis attacks, Tor only sends fixed-sized messages between relays, in the

[2] We configure OPUS to use only the former, as DONAR already enables redundancy mechanisms that are specific to the Tor network.

[3] While Tor advertises using $n = 1$ by default, it effectively uses $n = 2$ [62].

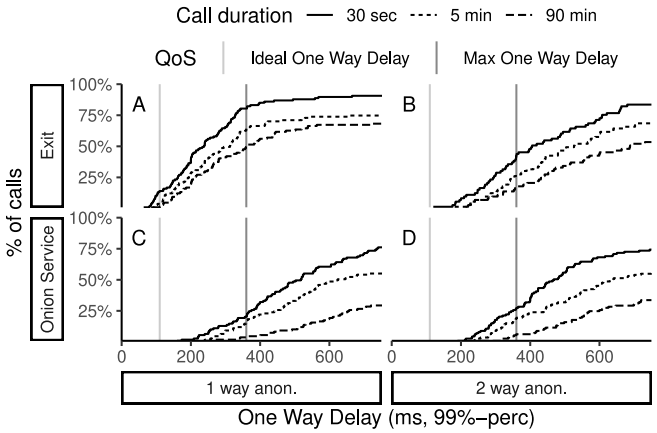

Figure 3: VoIP over a single Tor link: Evolution of the one-way delay's 99th percentile according to connection-link type and call duration.

form of 514-Byte *cells* [53, 59]. When a packet being transmitted over a Tor link is less than 514 Bytes in size, the Tor client pads it with random data.

3.1 Evaluation of Tor links' QoS

Tor is often described as a *low-latency* anonymization network. Its TCP streams over pre-established circuits enable, indeed, lower latency than anonymization networks where the relays for each message in a stream are chosen independently [12, 73, 76]. The latency of links in Tor, and in particular their stability, is however known to be unpredictable, which made several authors doubt Tor's ability to support low-latency applications such as VoIP [31, 66].

In this section, we report on our own experimental evaluation of the network QoS of Tor links. We confirm the observation made by other authors that a single Tor link is unsuitable for VoIP networking requirements as defined in the previous section. However, these measurements allow us to make the case for the foundational design choice in DONAR: using several, dynamically selected links.

We consider the following metrics: the connection stability, the variability of one-way latency, and the predictability of high latency from prior measurements. We use a load injector with varying packet-sending rates and, in order to measure one-way latency, a stub communication endpoint located on the same machine. The injector and the stub use two separate instances of the Tor client in its default configuration and create circuits independently. All reported experiments were conducted in January 2021.

Connection links. We start by analyzing how the two Tor modes, *Exit* and *Onion Services*, perform in terms of tail latency. Each of these modes can be declined in links providing either one-way or two-way anonymity. *Exit* links provide one-way anonymity by default but we can mimic two-way anonymity by making both caller and callee access the same public VoIP server. *Onion-Service* links provide two-way anonymity by default but we can reduce the number of relays and keep only one-way anonymity. We use the HIDDENSERVICESINGLEHOPMODE feature in the Tor daemon to achieve one-way anonymity over Onion Services.

Considering these 4 configurations, we simulated VoIP calls lasting 30 seconds, 5 minutes, and 90 minutes. The simulation strictly follows the requirements presented in **Section 2**. For each combination of configuration and call duration, we made 64 calls and present the results in Figure 3.

We start our analysis by focusing on Figure 3.A as it features the configuration on which Sharma *et al.* [70] base their claim that Tor links are suitable *as is* to support VoIP. With 37% of unacceptable calls (resp. 50%) for 5-minute (resp. 90-minute) calls, we argue the opposite. We identified three reasons explaining why our analysis differs. (1) They do not account for T_{frame} in their analysis. Since we use $T_{frame} = 40$ ms, our max acceptable latency is 360 ms. (2) They consider average latencies instead of the 99th percentile of their distribution. While we obtain similar average latencies, considering tail latency shows that 20% of calls suffer from unacceptable delays, even for short 30-second calls. (3) They consider only such 30-second calls when the average call duration is 3 minutes and when a significant share of calls last up to 90 minutes. Measuring links over a longer timespan shows, in fact, that latencies tend to increase with call duration.

Comparing the different configurations we observe that, in fact, no link type offers acceptable delays. We note (Figure 3.{B,D}) that the latency benefits from using the *Exit* mode mostly vanish when considering 2-way anonymity. Using one-way anonymity with the *Onion Service* mode (Figure 3.C) does not seem to improve tail latency; we presume this is because this feature is still experimental.

Moreover, not all link types are equal: using *Exit* links has two drawbacks. First, it requires the last relay of the circuit to hold the *Exit* tag. As *Exit* links can send data on the regular Internet, the last relay is particularly sensitive: only 25% of the relays accept to have this position. From the user's perspective, this situation eases de-anonymization attacks and, by limiting the scalability of the network, also harms performances. Moreover, using *Exit* links requires relaying traffic through an ad-hoc public server that must be trusted (e.g., Sharma *et al.* [70] use Mumble and Freeswitch PBX).

Considering that (i) no link type over Tor enables VoIP, and (ii) the *Exit* mode has severe limitations, we choose to focus solely on leveraging *Onion-Service* links to provide anonymous voice calls in the rest of this paper.

Connection stability. We evaluate the reliability of each Tor link type over our longest considered call duration (90 minutes). Figure 4 reports the cumulative rate of failed links (i.e., for which packets are no longer transmitted) as a function of time. After 10 minutes, all link types exhibit failure rates of at least 4%. The rate rises to between 7% and 16% after one hour. The failure difference between link types seems

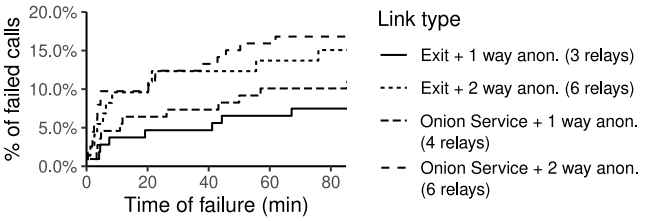

Figure 4: Failed Tor links over time.

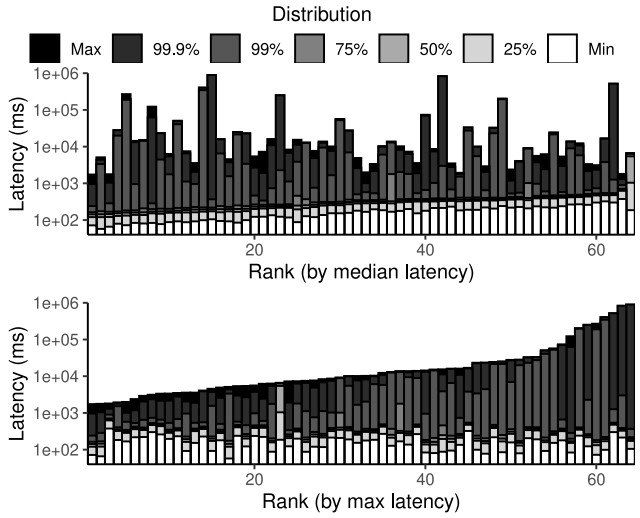

Figure 5: Tor links' latency distribution at 25 packet/sec ordered by median (top) and max (bottom) latency.

to be correlated with their number of relays: the more there are relays, the more likely failures are. None of the available links satisfies our QoS requirements: we need a solution that overcomes link breakage and allows calls to continue seamlessly.

Predictability of high latencies. The previous experiment shows that the distribution of latency across multiple links is highly skewed. We now evaluate if this skew results from a large number of poorly performing links with a few, identifiable, good links, or if any link can experience periodic latency bursts. Figure 5 presents the one-way-latency distribution for each of the 64 links, ranked by median latency (top) or max latency (bottom). There is no clear relationship between the general performance of a link and the occurrence of latency spikes. The maximal latency does not seem to depend much on the rest of the distribution and can reach very high values in all cases (often 3 times higher than the 75th percentile)[4]. We refer to these high latency periods as *latency spikes* in the rest of this paper.

Discussion. Our experiments confirm the general unpredictability of the performance of Tor links. Due to Tor's exclusive support for TCP[5], latency spikes for a single packet result in high latency for all following packets, delayed to be delivered in order—a phenomenon referred to as *head-of-line blocking*.

We observe, however, that the number of relays correlates with the probability of networking problems: larger numbers of relays are associated with higher failure rates or with latency spikes. We also note that most links provide good performance for a fraction of their use time, and failures across links do not seem to be correlated. As a result, we make the case for using multiple links, benefiting from periods of good performance, and quickly switching links when experiencing latency spikes.

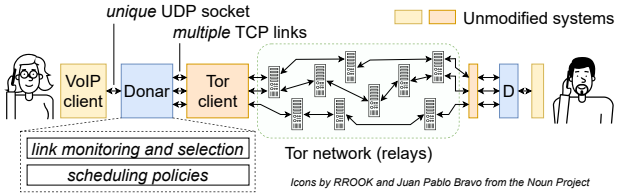

Figure 6: DONAR plays the role of a proxy between an unmodified VoIP application and the unmodified Tor client.

4 Donar: Enabling VoIP over Tor

DONAR operates as a proxy between a VoIP application and the Tor client, as illustrated in Figure 6. It does not require modifying either of the two systems. DONAR runs without any specific privileges; it only offers a UDP socket to the VoIP application's RTP protocol and opens TCP sockets (links) with the local Tor client. In conformance with our objective to make anonymous VoIP available with *readily-available* systems, we do not require the deployment of external support services. In particular, DONAR does not rely on the SIP signaling protocol but leverages instead Tor onion addresses to establish communication without leaking metadata about communicating parties.

Redundancy by piggybacking. DONAR leverages the fact that Tor only transmits data in the form of fixed-sized cells. Setting OPUS to the target bitrate of 32 kbps and using a sending period of 40 ms results in 172-Byte frames. The Tor client pads the remaining space with random data to reach a cell size of 514 Bytes. DONAR leverages, instead, this space to re-send the previous frame without changing the necessary bandwidth requirements[6]. Naturally, a redundant frame must

[4]This unpredictable performance is confirmed, in fact, by a blog post by the Tor project [63]. We quote: "*While adding more relays to the network will increase average-case Tor performance, it will not solve Tor's core performance problem, which is actually performance variance.*".

[5]TCP maps well to an efficient implementation of onion routing, i.e., it makes it possible to know when to create and dispose of circuits and it avoids the presence of packets that are untied to an existing circuit. UDP would also pose security challenges, e.g., enable DDoS attacks. The designers of Tor have clearly dismissed any support of UDP in Tor in the future [56].

[6]We are not limited to this configuration, and only require that the size of the frames emitted by the codec be less than half the available space minus the Tor headers (8 Bytes) and DONAR metadata (38 Bytes in the default configuration), i.e., less than 233 Bytes.

be sent on a different link than the first copy, to avoid head-of-line blocking between replicas. While redundant frames are subject to an additional T_{frame} latency (40 ms in the presented configuration), our rationale is that this latency combined with that of the link itself will still be lower than that of a link experiencing a latency spike. We detail next how we effectively enable link diversity.

Link Diversity. DONAR leverages multiple Tor Onion-Service links to multiplex traffic in two complementary ways. First, it spreads frame copies onto different links. This prevents packets containing subsequent frames from being subject to the same latency spike thereby arriving too late in a burst at the destination. This also lowers the load on each individual link (resulting, as shown in Section 3, in better availability). Second, DONAR ensures that the first and the second (redundant) copy of a given frame always travel on different links.

Enabling diversity requires (1) maintaining a set of open links and monitoring their performance; and (2) implementing a scheduling policy for selecting appropriate links for new packets. In the following, we detail these two aspects (§4.1 and §4.2) and complete the description of DONAR by detailing how calls are established (§4.3).

4.1 Link monitoring and selection

DONAR opens and monitors a set of Tor links and associates them with scores reflecting their *relative* latency performance. We start by detailing how latency scores are collected at the local client's side, and why they must also be collected from the remote client. We motivate our choice to classify links in performance groups, and how we dynamically select links in these groups throughout a call.

Measuring latency. Measuring transmission delays for packets sent over Tor is not straightforward. The RTP protocol uses UDP and does not send acknowledgments. We do not wish to add additional acknowledgment packets over Tor to measure round-trip times, as their padding in 514-Byte cells would double bandwidth consumption.

Rather than attempting to measure the *absolute* latencies of links, we leverage the use of multiple links to approximate their *relative* latency performance. Measures of performance are continuously collected on both sides of the communication, which we denote as node A and node B in the following. Local *aggregate* measures are then computed over a time window of duration w. We explore the impact of durations ranging from 0.2 to 32 seconds in our evaluation.

We base our measurements on an *out-of-order* metric for VoIP frames. This metric denotes, for an incoming frame f with sequence number i, the number of frames received *before* f with a higher sequence number than i. As TCP delivers packets in order, these frames necessarily travel on different links. For instance, if node A receives frame f with sequence number i from node B on link l after receiving frames with sequence numbers $i+1$, $i+2$, and $i+3$ on other links, we associate an out-of-order metric of 3 to frame f.

The local calculation of the out-of-order metric also applies to *missing* frames. Node A is aware of any *missing* frame f_m with a sequence number $i_m < i_c$ where i_c is the largest sequence number among all the frames received from node B. However, since the decision on which link a packet is sent is made by node B, it is not possible for node A to assign f_m's measurement to a specific link. To solve this problem, we include, in the DONAR headers in each packet, the list of links used for sending the latest n frames, where n is the maximum number of used links.

Nodes A and B must share their local aggregate measures to enable fast detection of latency spikes. Node A's local information about a link l approximates, indeed, the one-way latency from B to A, but not from A to B. Our experimental evaluation has shown that one-way latencies are highly consistent in both directions of a link, making node A's local estimation a good approximation also for the latency from A to B. However, this local approximation may be missing if the link has not been used recently by B to send packets to A. We alleviate this problem by embedding, in the DONAR metadata sent with each packet, the local aggregate measures for links that have been measured recently. Node A computes a final array of measures that include, for each link, either (1) the local aggregate measure only, if no remote aggregate was received; (2) the remote aggregate only, if the link was not recently used by B to send data to A; or (3) the average of these two measures if the link was used in both directions.

Link selection. Every w seconds, DONAR sorts links in decreasing order of aggregated scores over the last period and assigns links to three groups. The $L_{1\text{ST}}$ (first-class) group contains the $n_{1\text{ST}}$ *fastest* links. The $L_{2\text{ND}}$ (second-class) group contains the $n_{2\text{ND}}$ following links. Typically, we use the same number of links in the two groups, i.e., $n_{1\text{ST}} = n_{2\text{ND}}$. Finally, the remaining $n_{\text{INACTIVE}} = n_{\text{LINKS}} - n_{1\text{ST}} - n_{2\text{ND}}$ slowest links are assigned to the L_{INACTIVE} group.

The rationale for this classification is as follows. Links in the L_{INACTIVE} group experience sub-par performance and must remain idle. Links in the $L_{1\text{ST}}$ group have good performance and are invaluable in allowing fast delivery of VoIP packets. However, the number of good-performing links is limited at a given point in time, and using them systematically bears the risk of overloading them, resulting in lower performance and reliability (§3). Links in the $L_{2\text{ND}}$ group are less performant, but remain usable, and can reduce this risk of overload.

Links opening and maintenance. DONAR uses standard operations of the Tor client to open links. It lets the client select relays according to Tor rules. The client allows users to parameterize the number of used guard relays, as well as the length of the links (number of relays). DONAR leverages these parameters to enable different security/performance tradeoffs. We defer the discussion of strategies for setting these values and their security implications, to Section 5.

When starting a call, DONAR opens $n_{\text{LINKS}} = n_{1\text{ST}} + n_{2\text{ND}} + n_{\text{INACTIVE}}$ links and assigns them randomly to the three groups. When the Tor client notifies a link failure, DONAR simply requests a new link and assigns it to the L_{INACTIVE} group.

Links in the L_{INACTIVE} group will not be monitored locally. Some of these links may be associated with a remote score, but others will not be monitored on either side of the call. To enable *all* links to be monitored periodically, we implement a promotion and demotion mechanism between the $L_{2\text{ND}}$ and L_{INACTIVE} groups. When assigning links to groups at the end of a period of w seconds, DONAR picks the worst-performing link from the $L_{2\text{ND}}$ group and demotes it to the L_{INACTIVE} group. In return, it promotes to $L_{2\text{ND}}$ the link from the L_{INACTIVE} group that has been unused for the longest time.

4.2 Scheduling policies

The DONAR scheduler receives UDP RTP packets containing a single frame from the VoIP application. It first implements redundancy by using the pad space to piggyback packets that were previously sent on different links, then adds the necessary metadata, and finally creates a TCP packet to be sent onto one or two links from the $L_{1\text{ST}}$ and/or $L_{2\text{ND}}$ groups.

DONAR's default scheduling policy is named ALTERNATE. It sends each new packet to a *single* link. In doing so, it *alternates* between links from the $L_{1\text{ST}}$ and $L_{2\text{ND}}$ groups. This complies with the requirement to send the first and redundant copies of a frame on different links. DONAR picks the links from each group using a round-robin policy, thereby complying with the requirement of maximizing diversity.

We implement a second policy named DOUBLE-SEND. As the name implies, this policy selects *two* links—one from $L_{1\text{ST}}$ and one from $L_{2\text{ND}}$—for sending each new packet. Each frame is received four times: two as a primary copy, and two as a duplicate. This policy doubles the required bandwidth but has a higher chance to select a fast link for the primary copy of a frame, thereby reducing the risk of delivering the frame with an additional delay of T_{frame}. We note that the resulting bandwidth is the same as for TorFone [24]'s Duplication mode, which systematically sends VoIP packets onto the same two links.

4.3 Establishing communication

DONAR leverages Tor's mechanisms to allow callers and callees to establish a connection anonymously. Following our design goal of using only readily-available systems, we do not require the deployment of an existing or novel signaling protocol and, in particular, we do not use a SIP deployment. SIP requires, in fact, the use of trusted proxies and has been documented as leaking metadata to network observers [21,43]. Furthermore, with the exception of the audio codec negotiation, SIP functionalities largely overlap mechanisms already offered by Tor [21,43].

A caller can discover a callee by looking up a specific onion service identifier using the Tor DHT. This onion service identifier is obtained by other means, e.g., by using an anonymous chat service. The identifier can also be public while still preserving anonymity, as Tor prevents an external observer from determining that a specific client opens a circuit to a specific onion service. For instance, journalists could advertise an anonymous onion service for whistleblowers to use. We note that client-side authorization, as defined in the Tor rendezvous specification [54], could enable a callee to only allow calls from a whitelist of callers, but we leave the integration of this functionality to future work.

In the current DONAR implementation, the codec and its configuration are hardcoded. Codec and configuration negotiation require, unlike discovery, only communication between the two parties, and could employ a protocol similar to the subset of SIP dedicated to this task. We also leave this implementation to future work.

5 Security

DONAR leverages Tor without deploying additional infrastructure or modifying Tor itself. As a result, DONAR inherits the security assumptions and shortcomings of Tor. For instance, like Tor, DONAR does not provide protection from adversaries that can control the *entire* network [20,59] to perform traffic-correlation attacks [40,82]. Nevertheless, in terms of guarantees, it is reasonable to wonder whether DONAR worsens the security properties of Tor and to what extent.

In the definition of the so-called predecessor attack, Wright [82] observed that repeatedly creating new circuits causes clients to continuously degrade their security while increasing the probability that they will eventually choose a malicious relay as the first node of a circuit. Wright [81] proposed to address this problem by using what is now known as guards. Specifically, each Tor client chooses a small number of guards and uses them for all the circuits it ever creates. This suggests that DONAR's impact on security depends mainly on the fact that it can use a larger number of guards than the standard Tor implementation. We evaluate this impact from the perspective of three threats: (1) one endpoint deanonymizing the other, (2) an attacker that controls some relays or ASes and that tries to identify DONAR users, and (3) the same attacker deanonymizing both endpoints of a call and finally breaking anonymity.

Deanonymizing the other endpoint. According to the AnoA classification [6], sender/recipient anonymity refers to the ability to hide one endpoint's identity from the other. As discussed by Wright *et al.* [81], in a system with c corrupted relay nodes out of n and 1 guard per user, the probability of an endpoint's de-anonymizing the other is $\frac{c}{n}$. If we increase the number of guards to g, this probability becomes $1-(1-\frac{c}{n})^g$, which, for small values of $\frac{c}{n}$, can be approximated from above

by its first-order Taylor/Maclaurin expansion $g\frac{c}{n}$. Like most previous work, this analysis focuses on a random distribution of compromised guards. Adversaries can also leverage path selection algorithms to strategically place malicious guards and increase their probability of being selected although countermeasures exist [78].

Identifying DONAR users. Identifying a DONAR endpoint is equivalent to de-anonymizing any onion service, i.e., identifying which client node is reachable through this service. An adversary controlling a guard relay and knowing the onion address of a callee may observe traffic and employ traffic fingerprinting techniques [10,45,55,61,65] or use a fake DONAR client and perform timing attacks [58] to identify that a specific client is accepting DONAR calls. The use of several (g) guards in DONAR also increases the probability of this attack to $1-(1-\frac{c}{n})^g$, and thus by a factor of g for small values of $\frac{c}{n}$, like for the de-anonymization of one endpoint. This attack can however be mitigated by using the client-authorization feature offered by V3 Onion Services [54]. Finally, while several authors have shown that an adversary could locate onion service endpoints even when they were not publicly advertised [9,45,55,61], they have also proposed solutions to the Tor community.

De-anonymizing an ongoing call. To de-anonymize an ongoing call, an attacker needs to control guard nodes at both endpoints and employ traffic-correlation techniques [40]. As a result, like for the first two threats, the choice of the number of guards used by DONAR identifies a tradeoff between the likelihood of this attack and the performance of a call. In particular, since the attacker needs to control at least one guard on each side of the call, the associated probability grows from $(\frac{c}{n})^2$ with one guard to $(1-(1-\frac{c}{n})^g)^2$ with g guards. This implies that it grows even more slowly for small values of $\frac{c}{n}$ than the two previous probabilities.

Finally, we also observe that passive traffic correlation attacks turn out to be more difficult to perform when multiple calls are ongoing as DONAR's traffic patterns do not vary between different calls. In this case, a passive attack must observe the start and/or the end of a call to be effective.

DONAR security configurations.

As discussed above, increasing the number of guards improves performance but it also increases the attack surface. For this reason, DONAR implements three security configurations that strike different tradeoffs between privacy and performance, as illustrated in Figure 7. We emphasize that each configuration sets up the unmodified Tor client via its legacy API. DONAR systematically uses 12 links, but link settings are different in each configuration. The *Default* configuration provides a security strength similar to the legacy Tor client with default Tor link settings, i.e., each link has 6 relays, and each client employs only 2 guards.[7] The *2-hop* configuration

[7]Even though Tor's documentation discusses using only one guard, the default client uses two.

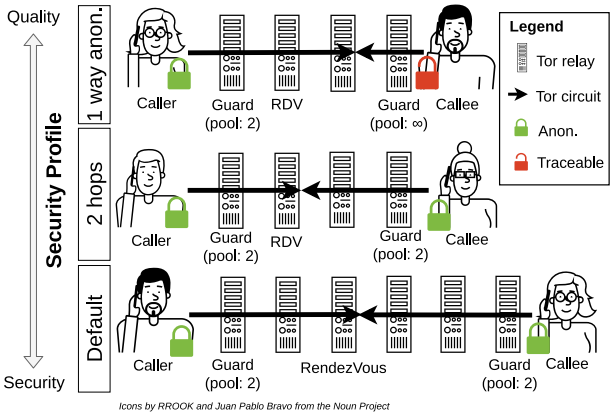

Figure 7: Security configurations.

sets up the Tor client so that each created link has two fewer Tor relays compared to Tor's default link settings. Finally, the *1-way-anonymity* configuration totally removes the anonymity of the callee using a *single* Tor relay (without the guard pool constraint) between the callee and the rendezvous point.

Security Discussion. Each of the threats we identified above relies on the control of at least one guard relay per affected endpoint. As discussed above, DONAR does not modify the guard configuration when providing anonymity for a user. Moreover, the use of guards decorrelates the number of links and the de-anonimization probability: using 12 links at once does not expose a user more than using only one. Additionally, compared to the *Default* configuration, the *2-hop* one reduces the number of relays in links by two. Decreasing the number of relays in links has been long debated in the Tor community. The main rationale for using 3-relay circuits (and thus 6-relay links) is that it makes it more difficult for an adversary that controls the last relay to identify the entry guard. On the other hand, an adversary can overcome this protection with relatively low investment in additional relays, and 3-relay circuits are more vulnerable to attacks based on denial of service [8]. These observations motivate our choice of 2-relay circuits with better latency in our *2-hop* configuration. Finally, the *1-way-anonymity* configuration does not provide anonymity to the callee but does not hamper the anonymity of the caller. Moreover, this mode is a standard feature of the Tor daemon that is used in production (e.g., by Facebook [1]).

Finally, we emphasize that DONAR users may also explore entirely different security configurations, by changing the number of Tor guards and/or relays for links, according to their own expected tradeoffs between performance and security.

6 Evaluation

DONAR is available open-source at https://github.com/CloudLargeScale-UCLouvain/Donar. The DONAR proxy interfaces a VoIP application with the Tor client.[8] We use two applications: (1) a configurable RTP emulator allowing a fine-grained control on the frames sent between parties, and running multiple occurrences of an experiment to study statistical variations; and (2) the actual `gstreamer` VoIP application using the OPUS codec. We deploy two isolated instances of either application on the same machine to accurately measure one-way delays for packets sent over Tor.

Tor's performance varies over time, with failures, disconnections, and latency spikes as identified in Section 3. Unless mentioned otherwise, we run each experiment a total of 64 times and present the distribution of results. We run the same configuration over a long time span, typically 5 hours, and also compare different configurations running in parallel.

6.1 Performance & comparison to SOTA

We start with the evaluation of the global performance of DONAR and its ability to meet the requirements summarized in Table 1. We use an audio stream of 32 kbps with a rate of 25 frames per second. We configure DONAR as follows: The window duration is $w = 2s$ and we open a total of $n_{\text{LINKS}} = 12$ links including $n_{\text{1ST}} = 3$ links, $n_{\text{2ND}} = 3$ links, and $n_{\text{INACTIVE}} = 6$ links. We present a comprehensive analysis of the influence of these parameters in Section 6.2.

We consider the six possible variants of DONAR using either of the two scheduling policies ALTERNATE and DOUBLE-SEND combined with one of the three security configurations (*Default*, *2 hops*, or *1-way anonymity*). In addition, we implement two approaches representing the state of the art. SIMPLE is the direct use of a single Tor link to transfer VoIP data. It represents our reference in terms of bandwidth usage for the ALTERNATE policy. TORFONE implements the duplication strategy used in TorFone [24]: It sends each new packet on two links, representing a reference for bandwidth usage for the DOUBLE-SEND policy.

No call interruption. We start by studying the percentage of dropped calls for all configurations. We run 96 instances of a 90-minute call for each combination and count the percentage of dropped calls. For SIMPLE, a broken Tor link invariably results in a dropped call. The DONAR variants and TORFONE, instead, re-establish broken links and thus consider their calls dropped whenever they miss 25 consecutive frames. Figure 8 presents the results. All DONAR variants perform better than the previous approaches and meet the goal of less than 2% of dropped calls. We only record, in fact, dropped calls for the most conservative of our setups, i.e., combining the ALTERNATE policy with the *default* configuration, and even then not exceeding 2%. TORFONE only meets the goal in the *1-way anonymity* configuration.

[8] The Tor software is evolving quickly, especially considering v3 onions. To benefit from latest bug fixes, we compiled Tor from branch `maint-0.4.4` (commit `09a1a34ad1`) and patch #256.

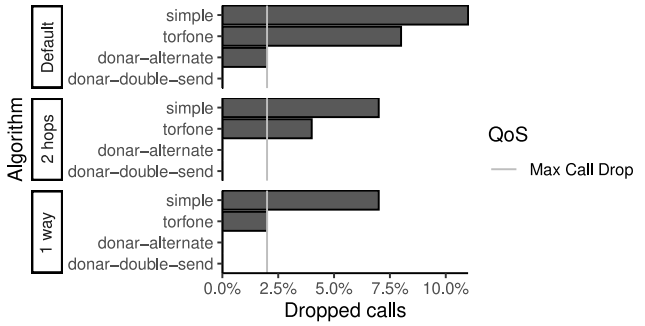

Figure 8: Dropped calls after 90 minutes for SIMPLE, TORFONE, and DONAR setups.

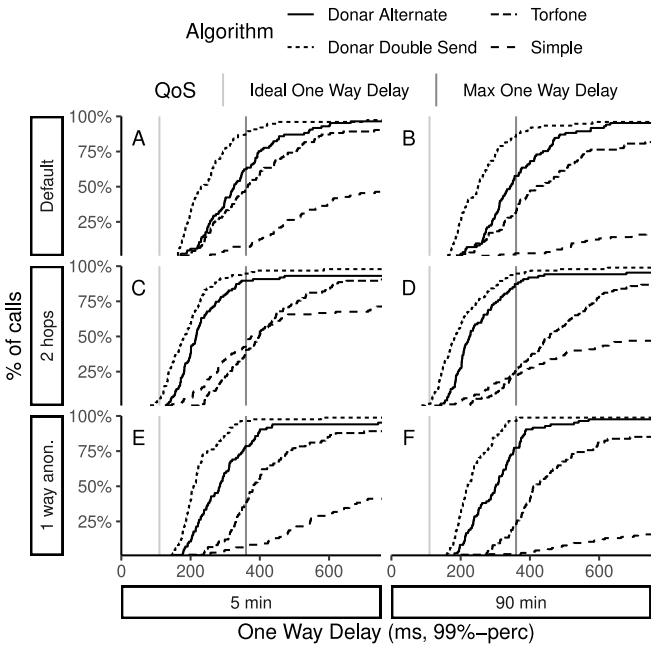

Figure 9: Latency comparison between SIMPLE, TORFONE, DONAR ALTERNATE and DONAR DOUBLE-SEND.

Interactive conversations & good voice quality. These objectives require a sufficient bitrate—met by using a 32 kbps bitrate in our experiments—and receiving at least 99% of VoIP frames within the maximum acceptable latency. The OPUS codec can, indeed, mask the loss of 1% of the frames with no perceptible quality degradation.

We present the distributions of frame delivery latencies in Figure 9. Our mouth-to-ear latency objective is 150 ms, and our limit is 400 ms. As T_{frame}=40 ms, we wish network delays for delivering frames to be of 110 to 360 ms. We use two vertical lines to denote these boundaries.

For all security policies and call durations, the DONAR DOUBLE-SEND algorithm provides at least 87% (*Default*, 90 minutes) of successful calls. Considering only our optimized security policies, the ratio of successful calls is even higher at

95%. These results must be compared to TORFONE, as both approaches send the same amount of data on the wire. TORFONE enables as low as 23% (*1-way anon.*, 90 minutes) and at most 47% (*Default*, 5 minutes) of successful calls. Compared to DONAR DOUBLE-SEND's worst performance (*Default*, 90-minute configuration), there is a 55-point difference with TORFONE in favor of DONAR.

Conversely, we observe that DONAR ALTERNATE does not fit all configurations: for its *Default* security policy, it enables only 62% (resp. 57%) of successful calls for 5 minutes (resp. 90 minutes). Results are better with *1-way anon.*: 78% (resp. 77%) for 5-minute (resp. 90-minute) calls. However, only the *2-hop* configuration seems to offer acceptable quality, enabling at least 87% of successful calls. Compared to the SIMPLE mode that sends the same amount of data, this is a 55-point gain points compared to DONAR's worst performance. With the *2-hop* configuration, it is a 43 points (resp. 65 points) for 5-minute (resp. 90-minute) calls improvement on SIMPLE.

To conclude, DONAR DOUBLE-SEND is able to offer a high ratio of successful calls in most situations (87%+ compared to 23%+ for TORFONE); it is a versatile solution at the cost of added redundancy on the wire. In comparison, DONAR ALTERNATE has no overhead but is way more sensitive to the configuration: it only works well with the *2-hop* security policy (87%+ compared to 46%+ for SIMPLE). With a difference of at most 4% between the 5-minute and 90-minute measurements, DONAR adds a new interesting property: latency stability over time. We argue that our two sending policies represent a significant improvement in terms of delay compared to the state of the art.

Using the `gstreamer` VoIP client. We experiment with the replay of an audio file using the `gstreamer` VoIP application. We collect statistics about its jitter buffer. `gstreamer` only allows a static-size jitter buffer. We configure this buffer based on our previous experiments, so as to absorb latencies between the minimum observed latency and the 99th-perc. latency, and count the number of calls that systematically meet latency requirements out of the 64 experiments done for each configuration. Our results confirm that DONAR DOUBLE-SEND is able to meet the 360 ms latency threshold for most experiments in all configurations, while the ALTERNATE policy works best under the *2-hop* configuration. We also confirmed empirically the results obtained under the *2-hop* configuration and the two scheduling policies by performing actual calls between two laptops: we could not detect any degradation in sound quality throughout any of the calls.

6.2 Microbenchmarks

In the following, we present an analysis of the influence of each of DONAR's parameters, and of the complementarity of its mechanisms. We focus on the six possible DONAR variants and, to factor out the impact of security configurations, we also consider a version of DONAR using 4 relays per link and an unlimited number of guards.

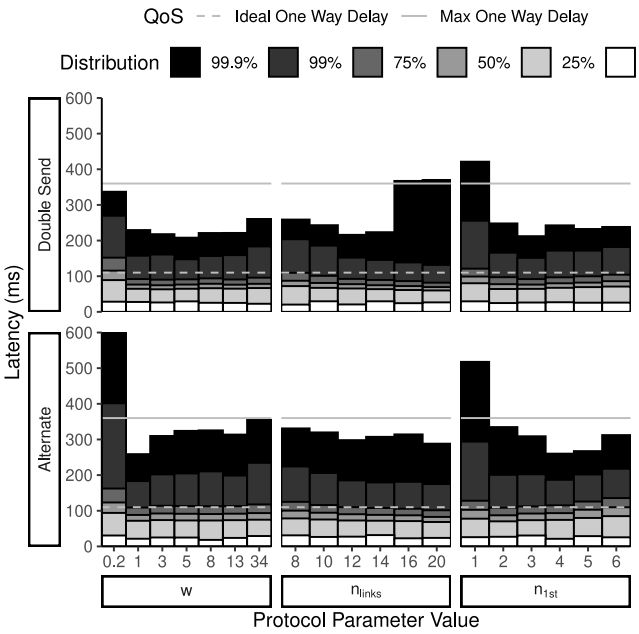

Figure 10: Impact of protocol parameters (w, n_{LINKS} and $n_{1\text{ST}} = n_{2\text{ND}}$) on frame delivery latencies.

Protocol parameters. DONAR has 3 main parameters: w, $n_{\text{LINKS}}, n_{1\text{ST}}$ (we use $n_{1\text{ST}} = n_{2\text{ND}}$). In the experiments reported in the previous section, we employed the default values of $w = 2s$, $n_{\text{LINKS}} = 12$, and $n_{1\text{ST}} = n_{2\text{ND}} = 3$. We detail in the following how we selected this default configuration.

We present, in Figure 10, an analysis of the influence of each parameter on the distribution of frame delivery latencies. Parameter w determines how far in the past we consider out-of-order metrics when computing link scores. It also determines how many times we need to probe a link before deciding to stop using it. A lower value of w enables a fast reaction at the risk of switching too many links with unreliable scores, while a larger value promotes links that are stable over time. We can observe on the left side of Figure 10 that the best value of w for the DOUBLE-SEND policy is 5s, while the best for ALTERNATE appears to be 2s. Additional benchmarks on the $[1,8]$ range with a smaller step led us to select the latter value as the default.

The n_{LINKS} parameter controls the total number of open links and, therefore, both the level of achievable diversity and the load of route maintenance on the Tor network. We evaluate n_{LINKS} values from 8 to 20. The ALTERNATE policy performs best with 20 links, while the DOUBLE-SEND policy performs best with 12 links. To limit the load on Tor, we select this latter value as the default.

Finally, parameter $n_{1\text{ST}} = n_{2\text{ND}}$ directly controls the number of links that are actively used to send packets. On the one hand, for a given value of n_{LINKS}, a small value of $n_{1\text{ST}}$ increases the likelihood of selecting only good-performing links. On the other hand, a large value increases diversity and the frame rate on each link, resulting in higher stability as we have shown

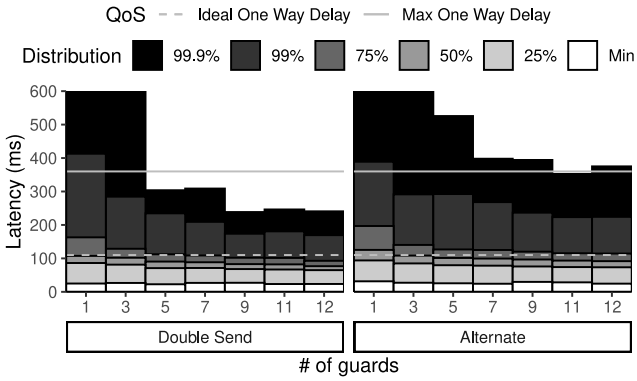

Figure 11: Impact of Tor guards number on latencies.

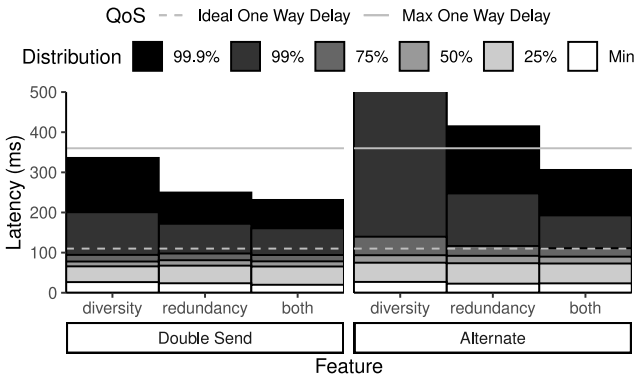

Figure 12: Diversity & redundancy complementarity.

in Section 3. Using $n_{1ST} = 1$ yields high latencies with either variant, while $n_{1ST} = 3$ or $n_{1ST} = 4$ offers a good compromise. We choose $n_{1ST} = 3$ as our default value.

Impact of the size of the guard pool. We considered using different sizes for the guard pool for the different security configurations detailed in Section 5. We further explore the impact of this parameter on DONAR's performance. Our results, shown in Figure 11, confirm that, in order to achieve the best latency, it is preferable to have as many guards as the number of links, in our case $n_{LINKS} = 12$. This number is, however, the result of a compromise with the attack surface. In our performance evaluation, we chose to stay conservative by not modifying the number of guards but we demonstrate here this choice has a cost in terms of performance.

Complementarity of diversity and redundancy. We analyze to which extent the two enabling mechanisms of DONAR, diversity and redundancy, contribute to its performance. We present in Figure 12 latency when using only link selection (diversity), using only redundancy by piggybacking, and using both. Activating both features is clearly beneficial for both scheduling policies, but, unsurprisingly, the impact of redundancy by piggybacking on high percentiles of the distribution is larger for the ALTERNATE strategy than for the DOUBLE-SEND strategy, as the latter enables redundancy by sending packets twice.

We further wish to understand how diversity and redundancy interact when used simultaneously, by analyzing, for each frame, which group of links delivers it for the first time, and whether this first delivery concerns a primary or a duplicate copy. The first delivery of a frame, indeed, results from a *race* between two send operations (with the ALTERNATE policy) and four send operations (with DOUBLE-SEND).

When using the ALTERNATE policy, 94% of the primary frame copies sent on a link of the L_{1ST} group arrive first. In 6% of the cases, the first copy that is received is the duplicate sent 40 ms later over an L_{2ND} link. When primary frame copies are sent over L_{2ND} links, however, only 48% arrive before the duplicate copy sent over an L_{1ST} link; 52% of the frames arrive first as the duplicate copy, despite the latter being sent 40 ms later. When using the DOUBLE-SEND policy, 73% of the frames are received first as a primary copy on the L_{1ST} link, 14% are received as a primary copy on an L_{1ST} link, and only 13% are received as a duplicate copy. Using L_{2ND} links remains useful. It provides more diversity, while still leveraging the reliability of the best links. Moreover, it decreases the load on each individual link, reducing the risk of performance degradation on each of them.

Link monitoring effectiveness. Appendix A presents a supplementary study of the effectiveness of link monitoring, where we analyze a trace of link classification and selection.

7 Related Work

VoIP over anonymization networks poses significant challenges as it combines the need for strong security with low and stable latency requirements. Three main families of anonymization networks have emerged: onion-route-, mix-net- and DC-net-based networks. The former do not protect from global adversaries that control the entire network whereas the latter two do.

The objective of DONAR is to leverage a readily-available system. Only two anonymity networks satisfy this requirement, Tor and Vuvuzela [76]. We discuss, nonetheless, the practicability of VoIP over a larger set of existing approaches, even if they are not effectively deployed.

Onion-route-based networks. Sharma *et al.* [70] called to re-think the feasibility of voice calling over Tor and claim that VoIP is feasible over Tor. However their analysis suffers from several shortcomings: they consider average latency instead of tail latency, they do their measurements only for 30 seconds, they do not evaluate dropped calls, they only provide one-way anonymity, etc. Karopoulos *et al.* [21, 43] explore the porting of SIP infrastructures on Tor. The main principle of their work is to preserve privacy in the SIP signaling protocol, in contrast with DONAR that leverages Tor's built-in mechanisms for establishing calls. The RTP stream is transmitted using a single Tor onion link. This approach behaves like SIMPLE from our experimental evaluation in this respect. TorFone [24] tries to improve latency by duplicating traffic over only two onion

links without any scheduling and monitoring mechanisms. As demonstrated in Section 6, the TORFONE policy is not sufficient to meet VoIP requirements.

Mix-net-based networks. Mix networks [13] batch and shuffle packets via *mix nodes* to prevent attackers from performing global traffic analysis. However, in doing so, they inherently incur high latency, which makes them unusable in latency-sensitive applications. A key solution to reduce packet delivery times consists in using cover traffic to prevent the mixes from having to wait too long before having enough packets to send a batch. Accordingly, the challenge faced by the latest research on mix-nets, such as Karaoke [48], Vuvuzela [76], Riffle [46], Loopix [64], Aqua [51], and Stadium [73], consists in designing an adequate mix-net with the best tradeoff between minimizing the necessary cover traffic while guaranteeing good resilience to traffic analysis. In their best-case usage scenario, these approaches drastically reduce latency from several hundred seconds to a few seconds, but this remains very far from VoIP requirements.

DC-net-based networks. Latency can be reduced by avoiding batching. Instead of using mix nodes, Dining-Cryptographer Networks (DC-nets) rely on anonymous broadcast among all network participants [13]. DC-nets have two inherent shortcomings: (i) they incur a high bandwidth overhead, i.e., the number of messages exchanged to send one message anonymously grows quadratically with the number of network participants, and (ii) they are vulnerable to denial of service attacks from malicious participants that can jam the whole network. Being resistant to such attacks requires, for instance, the use of zero-knowledge proofs to detect misbehavior but this is very costly in computation and results in increased delivery latency [25]. Consequently, a number of research works on DC-nets have emerged in recent years. Dissent [16, 80], Riposte [15], and Verdict [17] resist jamming attacks while trying to provide the best tradeoff between reducing the number of exchanged messages (e.g by splitting the network into smaller parts) and the impact of computational cost on latency. However, despite their efforts, their latency remains far too high for VoIP and increases with the number of users.

Anonymization networks designed for VoIP. Herd [50] is based on the mix-net principle. It was specifically designed for VoIP. Its hybrid approach uses mix nodes along with super peers organized in trust zones. Herd can provide VoIP on its anonymity network with good resistance against global adversaries. Its evaluation shows expected latency values of 400 ms. The recent work on Yodel [49] removes the concept of trust zones and supports higher percentages of dishonest nodes than Herd. However, this comes at the cost of latency increasing with the probability of having dishonest mix nodes. For instance, in a Tor-like environment (i.e., $\sim$ 20% of malicious servers) latency already reaches $\sim$ 900 ms. To counterbalance this latency, Yodel uses a codec with poorer quality than OPUS. Even if both Herd and Yodel are promising designs, neither is currently deployed. Today's whistleblowers are, therefore, unable to communicate using these systems. Moreover, we point out that the evaluation of both systems has been performed in optimal conditions, and their performance in settings comparable to Tor's deployment remains unstudied. For instance, Yodel is evaluated on 100 powerful Amazon EC2 servers with no external interference. DONAR, on the other hand, satisfies VoIP latency requirements, even if Tor constantly relays traffic generated by over 2 million daily users. To summarize, Tor and Vuvuzela represent the only anonymization networks that are readily available and widely deployed today. Since Vuvuzela cannot support VoIP due to its high latency, DONAR over Tor represents the only solution that enables privacy-conscious users to communicate anonymously using VoIP and with a good QoE.

Latency improvements on Tor. We also reviewed existing proposals to improve latency in Tor. This latency depends on two main factors: (i) queuing delays (time spent in a relay), and (ii) transmission delays (time spent on the "wire", between two Tor relays). Ting [11] and LASTor [2] both reduce transmission delays by modifying the path selection algorithm. However, latency spikes are due to queuing delays [19], particularly because Tor does not perform any centralized load balancing of traffic. To reduce queuing delays, improved traffic scheduling policies have been integrated in the latests versions of Tor [38, 39], but we still observe latency spikes. Alternative path selection algorithms use historical data on relay performance [71, 79] or probe circuits upon their creation [5]. They are inefficient for VoIP as latency spikes are ephemeral; predictions are outdated after a few seconds.

Multipath. We are not the first to advocate for multipath. MORE [47] proposes to route independently each cell, but it is not designed to be used with circuits like in Tor. MPTCP [22, 23, 29] aggregates TCP links over multiple network interfaces. However, it makes assumptions (e.g., latency is independent of traffic) that do not hold over Tor. In response, dedicated multipath protocols specially tailored for onion routing networks [3, 18, 42, 83] emerged. Nevertheless, compared to DONAR, none of these approaches optimize tail latency as required by all real-time protocols, including VoIP.

8 Conclusion

We presented DONAR, a solution for readily-available, anonymous, and high-quality VoIP calls using the challenging but existing Tor network. DONAR circumvents Tor's inability to support the networking requirements of VoIP by sending audio frames over a diversity of links and using redundancy. It offers different tradeoffs between performance and security and successfully enables high-quality VoIP calls, e.g., with latency below 360ms during an entire 90-minute call.

Acknowledgments: We are thankful to the anonymous reviewers and to our shepherd, Harsha V. Madhyastha, for their constructive feedback. This work was partially funded by the O'Browser ANR grant (ANR-16-CE25-0005-03).

References

[1] Tor project issue trackers: Facebook's onion site is a single hop onion, but clicking on the Tor onion icon shows that it is a 6 hop circuit (issue #23875). https://gitlab.torproject.org/legacy/trac/-/issues/23875.

[2] Masoud Akhoondi, Curtis Yu, and Harsha V Madhyastha. LASTor: A low-latency AS-aware Tor client. In *IEEE Symposium on Security and Privacy*, S&P, 2012.

[3] Mashael AlSabah, Kevin Bauer, Tariq Elahi, and Ian Goldberg. The path less travelled: Overcoming Tor's bottlenecks with traffic splitting. In *International Symposium on Privacy Enhancing Technologies*, PETS. Springer, 2013.

[4] Soren Vang Andersen, Alan Duric, Henrik Astrom, Roar Hagen, W. Bastiaan Kleijn, and Jan Linden. Internet Low Bit Rate Codec (iLBC). Request for Comments (RFC) 3951, Internet Engineering Task Force (IETF), December 2004.

[5] Robert Annessi and Martin Schmiedecker. Navigator: Finding faster paths to anonymity. In *European Symposium on Security and Privacy*, EuroS&P. IEEE, 2016.

[6] Michael Backes, Aniket Kate, Praveen Manoharan, Sebastian Meiser, and Esfandiar Mohammadi. AnoA: A framework for analyzing anonymous communication protocols. In *26th Computer Security Foundations Symposium*, CSF. IEEE, 2013.

[7] Luca Barbato. RTP payload format for Vorbis encoded audio. Request for Comments (RFC) 5215, Internet Engineering Task Force (IETF), August 2008.

[8] Kevin Bauer, Joshua Juen, Nikita Borisov, Dirk Grunwald, Douglas Sicker, and Damon McCoy. On the optimal path length for Tor. In *3rd Hot Topics in Privacy Enhancing Technologie*, HotPets, 2010.

[9] Alex Biryukov, Ivan Pustogarov, and Ralf-Philipp Weinmann. Trawling for tor hidden services: Detection, measurement, deanonymization. In *Symposium on Security and Privacy*, S&P. IEEE, 2013.

[10] Xiang Cai, Xin Cheng Zhang, Brijesh Joshi, and Rob Johnson. Touching from a distance: Website fingerprinting attacks and defenses. In *ACM conference on Computer and communications security*, CCS, 2012.

[11] Frank Cangialosi, Dave Levin, and Neil Spring. Ting: Measuring and exploiting latencies between all tor nodes. In *Internet Measurement Conference*, IMC, 2015.

[12] David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2), 1981.

[13] David L. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of cryptology*, 1(1), 1988.

[14] Opus Codec. Codec landscape. https://opus-codec.org/comparison/, 2020.

[15] Henry Corrigan-Gibbs, Dan Boneh, and David Mazières. Riposte: An anonymous messaging system handling millions of users. In *Symposium on Security and Privacy*, S&P. IEEE, 2015.

[16] Henry Corrigan-Gibbs and Bryan Ford. Dissent: accountable anonymous group messaging. In *17th ACM conference on Computer and communications security*, CCS, 2010.

[17] Henry Corrigan-Gibbs, David Isaac Wolinsky, and Bryan Ford. Proactively accountable anonymous messaging in Verdict. In *22nd USENIX Security Symposium*, 2013.

[18] Wladimir De la Cadena, Daniel Kaiser, Asya Mitseva, Andriy Panchenko, and Thomas Engel. Analysis of multi-path onion routing-based anonymization networks. In *IFIP Annual Conference on Data and Applications Security and Privacy*, DBSec. Springer, 2019.

[19] Prithula Dhungel, Moritz Steiner, Ivinko Rimac, Volker Hilt, and Keith W Ross. Waiting for anonymity: Understanding delays in the Tor overlay. In *10th International Conference on Peer-to-Peer Computing*, P2P. IEEE, 2010.

[20] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, Naval Research Lab Washington DC, 2004.

[21] Alexandros Fakis, Georgios Karopoulos, and Georgios Kambourakis. OnionSIP: Preserving privacy in SIP with onion routing. *J. Univers. Comput. Sci.*, 23(10), 2017.

[22] Alexander Froemmgen, Jens Heuschkel, and Boris Koldehofe. Multipath TCP scheduling for thin streams: Active probing and one-way delay-awareness. In *International Conference on Communications*, ICC. IEEE, 2018.

[23] Alexander Frommgen, Tobias Erbshäußer, Alejandro Buchmann, Torsten Zimmermann, and Klaus Wehrle. ReMP TCP: Low latency multipath TCP. In *International Conference on Communications*, ICC. IEEE, 2016.

[24] Van Gegel. TORFone: secure VoIP tool. http://torfone.org/, 2013.

[25] Philippe Golle and Ari Juels. Dining cryptographers revisited. In *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2004.

[26] GStreamer. Gstreamer: open source multimedia framework. https://gstreamer.freedesktop.org/, 2020.

[27] Ben Doherty (The Guardian). Vodafone australia admits hacking fairfax journalist's phone. https://www.theguardian.com/business/2015/sep/13/vodafone-australia-admits-hacking-fairfax-journalists-phone, 2015.

[28] Yi Han, Damien Magoni, Patrick Mcdonagh, and Liam Murphy. Determination of bit-rate adaptation thresholds for the opus codec for voip services. In *Symposium on Computers and Communications*, ISCC. IEEE, 2014.

[29] Mark Handley, Olivier Bonaventure, Costin Raiciu, and Alan Ford. TCP extensions for multipath operation with multiple addresses. Request for Comments (RFC) 6824, Internet Engineering Task Force (IETF), January 2013.

[30] G. Herlein, J. Valin, A. Heggestad, and A. Moizard. RTP payload format for the Speex codec. Request for Comments (RFC) 5574, Internet Engineering Task Force (IETF), June 2009.

[31] Stephan Heuser, Bradley Reaves, Praveen Kumar Pendyala, Henry Carter, Alexandra Dmitrienko, William Enck, Negar Kiyavash, Ahmad-Reza Sadeghi, and Patrick Traynor. Phonion: Practical protection of metadata in telephony networks. *Proceedings on Privacy Enhancing Technologies*, 2017(1), 2017.

[32] Christian Hoene, Jean-Marc Valin, Koen Vos, and Jan Skoglund. Summary of Opus listening test results. https://tools.ietf.org/html/draft-ietf-codec-results-03, 2013.

[33] Jan Holub, Michael Wallbaum, Noah Smith, and Hakob Avetisyan. Analysis of the dependency of call duration on the quality of VoIP calls. *IEEE Wireless Communications Letters*, 7(4):638–641, 2018.

[34] Monty Icenogle. T-mobile does have a hard 4 hour single call duration limit. https://kd6cae.livejournal.com/271120.html, 2015.

[35] ITU. ITU-T recommendation G.114, "one way transmission time". https://www.itu.int/rec/T-REC-G.114, 2003.

[36] ITU. E.800 : Definitions of terms related to quality of service, 2008.

[37] ITU. G.1028: End-to-end quality of service for voice over 4G mobile networks. https://www.itu.int/rec/T-REC-G.1028, 2019.

[38] Rob Jansen, John Geddes, Chris Wacek, Micah Sherr, and Paul Syverson. Never been KIST: Tor's congestion management blossoms with kernel-informed socket transport. In *23rd USENIX Security Symposium*, 2014.

[39] Rob Jansen, Matthew Traudt, John Geddes, Chris Wacek, Micah Sherr, and Paul Syverson. Kist: Kernel-informed socket transport for Tor. *ACM Transactions on Privacy and Security (TOPS)*, 22(1):1–37, 2018.

[40] Aaron Johnson, Chris Wacek, Rob Jansen, Micah Sherr, and Paul Syverson. Users get routed: Traffic correlation on Tor by realistic adversaries. In *ACM SIGSAC conference on Computer & communications security*, CCS, 2013.

[41] kamedo2. Results of the public multiformat listening test. https://listening-test.coresv.net/results.htm, 2014.

[42] Hasan T Karaoglu, Mehmet Burak Akgun, Mehmet Hadi Gunes, and Murat Yuksel. Multi path considerations for anonymized routing: Challenges and opportunities. In *5th International Conference on New Technologies, Mobility and Security*, NTMS. IEEE, 2012.

[43] Georgios Karopoulos, Alexandros Fakis, and Georgios Kambourakis. Complete SIP message obfuscation: PrivaSIP over Tor. In *9th International Conference on Availability, Reliability and Security*. IEEE, 2014.

[44] Byeong Hoon Kim, Hyoung-Gook Kim, Jichai Jeong, and Jin Young Kim. VoIP receiver-based adaptive playout scheduling and packet loss concealment technique. *IEEE Transactions on consumer Electronics*, 59(1):250–258, 2013.

[45] Albert Kwon, Mashael AlSabah, David Lazar, Marc Dacier, and Srinivas Devadas. Circuit fingerprinting attacks: Passive deanonymization of Tor hidden services. In *24th USENIX Security Symposium*, 2015.

[46] Albert Kwon, David Lazar, Srinivas Devadas, and Bryan Ford. Riffle: An efficient communication system with strong anonymity. *Proceedings on Privacy Enhancing Technologies*, 2016(2):115–134, 2016.

[47] Olaf Landsiedel, Alexis Pimenidis, Klaus Wehrle, Heiko Niedermayer, and Georg Carle. Dynamic multipath

onion routing in anonymous peer-to-peer overlay networks. In *IEEE Global Telecommunications Conference*, GlobeCom, 2007.

[48] David Lazar, Yossi Gilad, and Nickolai Zeldovich. Karaoke: Distributed private messaging immune to passive traffic analysis. In *13th USENIX Symposium on Operating Systems Design and Implementation*, OSDI, 2018.

[49] David Lazar, Yossi Gilad, and Nickolai Zeldovich. Yodel: strong metadata security for voice calls. In *27th ACM Symposium on Operating Systems Principles*, SOSP, 2019.

[50] Stevens Le Blond, David Choffnes, William Caldwell, Peter Druschel, and Nicholas Merritt. Herd: A scalable, traffic analysis resistant anonymity network for voip systems. In *ACM Conference on Special Interest Group on Data Communication*, SIGCOMM, 2015.

[51] Stevens Le Blond, David Choffnes, Wenxuan Zhou, Peter Druschel, Hitesh Ballani, and Paul Francis. Towards efficient traffic-analysis resistant anonymity networks. *ACM SIGCOMM Computer Communication Review*, 43(4):303–314, 2013.

[52] Yi J Liang, Nikolaus Farber, and Bernd Girod. Adaptive playout scheduling and loss concealment for voice communication over IP networks. *IEEE Transactions on Multimedia*, 5(4):532–543, 2003.

[53] Zhen Ling, Junzhou Luo, Wei Yu, and Xinwen Fu. Equal-sized cells mean equal-sized packets in Tor? In *International Conference on Communications*, ICC. IEEE, 2011.

[54] Nick Mathewson, George Kadianakis, David Goulet, Tim Wilson-Brown, Hans-Christoph Steiner, Filipo Valsorda, and Roger Dingledine. Tor rendezvous specification - version 3, 2017.

[55] Prateek Mittal, Ahmed Khurshid, Joshua Juen, Matthew Caesar, and Nikita Borisov. Stealthy traffic analysis of low-latency anonymous communication using throughput fingerprinting. In *18th ACM conference on Computer and communications security*, CCS, 2011.

[56] Nick Montfort, Arthur Edelstein, Robert Ransom, and Yawning Angel. Tor project feature tracker: Closed enhancement, "UDP over Tor". https://trac.torproject.org/projects/tor/ticket/7830, 2013.

[57] Sue B Moon, Jim Kurose, and Don Towsley. Packet audio playout delay adjustment: performance bounds and algorithms. *Multimedia systems*, 6(1):17–28, 1998.

[58] Steven J Murdoch and George Danezis. Low-cost traffic analysis of Tor. In *Symposium on Security and Privacy*, S&P. IEEE, 2005.

[59] Steven J Murdoch, Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router (2013 DRAFT v1). https://gitweb.torproject.org/tor-design-2012.git/, 2014.

[60] David Kaplan (Newsweek). Suspicions and spies in silicon valley. https://www.newsweek.com/suspicions-and-spies-silicon-valley-109827, 2006.

[61] Andriy Panchenko, Asya Mitseva, Martin Henze, Fabian Lanze, Klaus Wehrle, and Thomas Engel. Analysis of fingerprinting techniques for Tor hidden services. In *Workshop on Privacy in the Electronic Society*, 2017.

[62] Mike Perry. The move to two guard nodes. https://gitweb.torproject.org/user/mikeperry/torspec.git/tree/proposals/xxx-two-guard-nodes.txt?h=twoguards, 2018.

[63] Mike Perry. Tor's open research topics: 2018 edition | tor blog. https://blog.torproject.org/tors-open-research-topics-2018-edition, 2018.

[64] Ania M Piotrowska, Jamie Hayes, Tariq Elahi, Sebastian Meiser, and George Danezis. The Loopix anonymity system. In *26th USENIX Security Symposium*, 2017.

[65] Tobias Pulls and Rasmus Dahlberg. Website fingerprinting with website oracles. *Proceedings on Privacy Enhancing Technologies*, 2020(1):235–255, 2020.

[66] Maimun Rizal. *A Study of VoIP performance in anonymous network-The onion routing (Tor)*. PhD thesis, Niedersächsische Staats-und Universitätsbibliothek Göttingen, 2014.

[67] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. Request for Comments (RFC) 3261, Internet Engineering Task Force (IETF), June 2002.

[68] Katrin Schoenenberg, Alexander Raake, Sebastian Egger, and Raimund Schatz. On interaction behaviour in telephone conversations under transmission delay. *Speech Communication*, 63:1–14, 2014.

[69] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. Request for Comments (RFC) 3550, Internet Engineering Task Force (IETF), July 2003.

[70] Piyush Kumar Sharma, Shashwat Chaudhary, Nikhil Hassija, Mukulika Maity, and Sambuddho Chakravarty. The road not taken: re-thinking the feasibility of voice calling over Tor. *Proceedings on Privacy Enhancing Technologies*, 2020(4):69–88, 2020.

[71] Robin Snader and Nikita Borisov. A tune-up for Tor: Improving security and performance in the Tor network. In *16th Annual Network & Distributed System Security Symposium*, NDSS, 2008.

[72] Tim Terriberry and Koen Vos. Definition of the Opus audio codec, 2012.

[73] Nirvan Tyagi, Yossi Gilad, Derek Leung, Matei Zaharia, and Nickolai Zeldovich. Stadium: A distributed metadata-private messaging system. In *26th Symposium on Operating Systems Principles*, SOSP. ACM, 2017.

[74] JM. Valin and K. Vos. Updates to the Opus Audio Codec. RFC 8251, October 2017.

[75] JM. Valin, K. Vos, and T. Terriberry. Definition of the Opus Audio Codec. Request for Comments (RFC) 6716, September 2012.

[76] Jelle Van Den Hooff, David Lazar, Matei Zaharia, and Nickolai Zeldovich. Vuvuzela: Scalable private messaging resistant to traffic analysis. In *25th Symposium on Operating Systems Principles*, SOSP, 2015.

[77] Voyced. Is there a maximum call length or duration. https://www.voyced.eu/clients/index.php/knowledgebase/397/Is-there-a-maximum-Call-length-or-duration.html, 2019.

[78] Gerry Wan, Aaron Johnson, Ryan Wails, Sameer Wagh, and Prateek Mittal. Guard placement attacks on path selection algorithms for Tor. *Proceedings on Privacy Enhancing Technologies*, 2019(4):272–291, 2019.

[79] Tao Wang, Kevin Bauer, Clara Forero, and Ian Goldberg. Congestion-aware path selection for tor. In *International Conference on Financial Cryptography and Data Security*, FC. Springer, 2012.

[80] David Isaac Wolinsky, Henry Corrigan-Gibbs, Bryan Ford, and Aaron Johnson. Dissent in numbers: Making strong anonymity scale. In *10th USENIX Symposium on Operating Systems Design and Implementation*, OSDI, 2012.

[81] Matthew Wright, Micah Adler, Brian N. Levine, and Clay Shields. Defending anonymous communications against passive logging attacks. In *IEEE Symposium on Security and Privacy*, S&P, page 28, USA, 2003. IEEE Computer Society.

[82] Matthew K Wright, Micah Adler, Brian Neil Levine, and Clay Shields. The predecessor attack: An analysis of a threat to anonymous communications systems. *ACM Transactions on Information and System Security*, 7(4):489–522, 2004.

[83] Lei Yang and Fengjun Li. mtor: A multipath Tor routing beyond bandwidth throttling. In *2015 IEEE Conference on Communications and Network Security*, CNS. IEEE, 2015.

A Appendix: Link monitoring effectiveness

We provide in this appendix a supplementary study of the effectiveness of link monitoring, dynamic link classification, and link selection. In particular, we assess whether link classification and selection reflect the behaviors discussed in Section 3.

We start by observing the distribution, over 64 calls, of the number of links that were classified as L_{1ST} *at least* once through the duration of a 90-minute call. This distribution is depicted in Figure 13. Note that we do not consider the first 40 seconds of each call, as DONAR has to bootstrap the process with random scores, and poorly-performing links could be assigned to the L_{1ST} group during this bootstrap. Between 6 and 12 links per call have been considered at least once in the L_{1ST} group in every call, with a majority of 8 to 10 links selected. This confirms our analysis that there is no single link that is consistently performing well in Tor, and that link performance varies significantly over time: Links that are poorly performing at a given time may be the best ones a few minutes later.

We study, in finer detail, the stability of links over time, focusing on a single call using the ALTERNATE policy with the Default configuration. We represent the latency of the first delivery of each frame in the first plot of Figure 14. This is the latency that is observed by the VoIP application. Latency remains low throughout the call. In the second plot, we decompose the latency of frames received on the L_{1ST} and L_{2ND} groups, including the first and second receptions. We can clearly see that the latency of the links in the L_{1ST} group is generally lower, and that outlier values are compensated by lower latency on a link in the L_{2ND} group. The third plot represents the assignment of the 12 links to link groups over

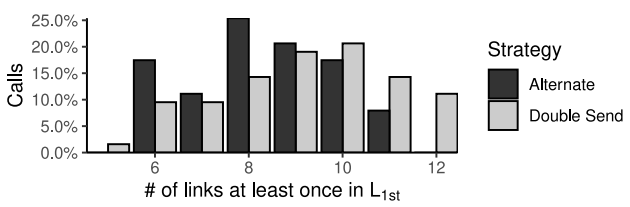

Figure 13: How many links were L_{1ST} at least once?

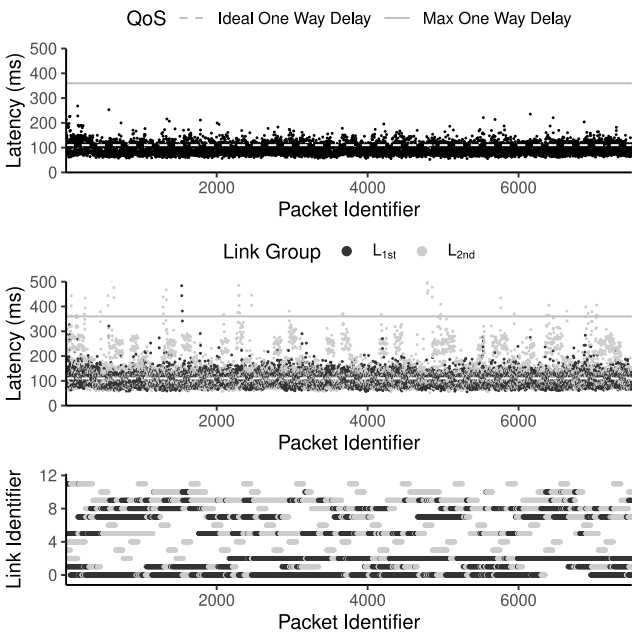

Figure 14: Stability over time.

time. We note that there was no link failure (and therefore no link replacement) in this experiment. Link 0 is, for instance, classified in L_{1ST} for a large part of the call, but suffers a latency spike around frame 6,500 and is rapidly classified in the $L_{INACTIVE}$ group. Link 2, initially in $L_{INACTIVE}$, is promoted 3 times with no effect to the L_{2ND} group, before being selected as L_{1ST} after its fourth promotion. Links 1, 5, 7 and 8 have highly heterogeneous behaviors, while links 3, 4, 6, 11 and 12 have consistently bad behaviors, and only appear in the L_{2ND} group upon their promotion before being quickly deactivated. While these links could be proactively replaced by opening new links, we do not deem it necessary and choose not to impose further link setup load on the Tor network.