Metadata-Private Communication for the 99%

Yossi Gilad
MIT and Boston University

ABSTRACT

In 2019, most Internet communication is encrypted [10]. However, it is still difficult to hide the communication metadata, like who communicates with whom and at what time, which usually remains exposed to anyone able to observe network traffic. Metadata reveals a great deal of information. Ex-government officials have stated that "if you have enough metadata you don't really need content," and "we kill people based on metadata" [13, 28]. Recent research proposes new systems that hide metadata. However, these solutions usually fall short on performance. Poor performance limits applications to a relatively small user-base, to run on desktop machines, and to means of communication with low requirements on latency such as e-mail exchange and text messaging.

In this article, we ask: are these limitations inherent? We sketch the requirements from the underlying communication system that are needed to facilitate metadata-private communication for several types of applications with a large user-base. We show how performance requirements from applications and restrictions of power consumption and network use on mobile devices may limit the security and privacy properties that the communication system can achieve. Finally, we motivate why meaningful metadata privacy guarantees are still tangible for many types of applications with a large user-base.

CCS CONCEPTS

• **Security and privacy → Distributed systems security**;

KEYWORDS

Security and privacy, distributed systems

ACM Reference Format:
Yossi Gilad. 2019. Metadata-Private Communication for the 99%. In *Proceedings of Communications of the ACM (Accepted to CACM).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Online privacy is a prominent topic in the news and receives growing attention from the public. This motivated messaging services such as WhatsApp, Signal, and Telegram to deploy end-to-end encryption, which hides the content of messages from anyone who listens on the communication. While encryption is widely deployed, it does not hide metadata: anyone capable of tapping the network links can learn who is communicating with whom, at what times and study their traffic volumes. Metadata reveals a lot about the underlying content. Public announcements by ex-government officials as well as the leaked Snowden documents have made it clear that intelligence organizations have a substantial interest in metadata even for encrypted communication since it often obviates the need for the actual content [13, 24, 28].

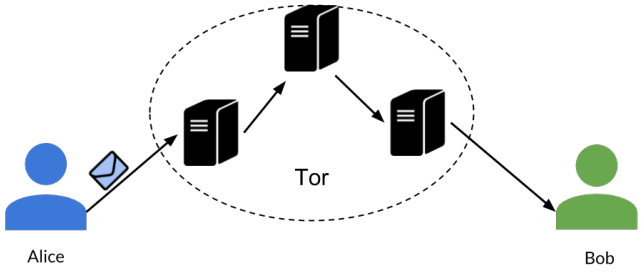

Figure 1: Alice sends a message to Bob through Tor. The message routes through three relays to hide its metadata.

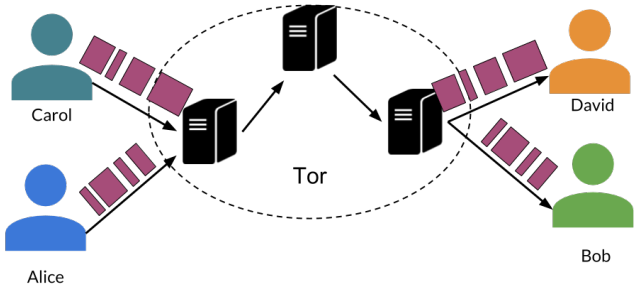

Figure 2: Eavesdropping on the communication links allows to identify which pairs of users communicate through Tor by correlating their traffic patterns.

The most popular system for hiding metadata, called Tor [7], routes user traffic through a series of relay servers as illustrated in Figure 1. In this fashion, the first relay only sees traffic to and from Alice, but it does not observe the other end of the conversation. Similarly, the last relay sees Bob's traffic but does not observe the user at the other end. So even if just one of the relays is honest, i.e., keeps secret which incoming message maps to what outgoing message that it forwards, the connection between Alice and Bob remains hidden. One of the key reasons for Tor's popularity is its performance, which can support mobile and desktop users and a variety of applications, such as messaging, web surfing, and VoIP calls. However, Tor is vulnerable to attackers that can observe traffic going in and out of the honest relays. By tapping relays, attackers can correlate messages that a relay receives to those that it sends, and follow a message from its source to destination. In fact, it is sufficient to tap the first and last Tor relays to correlate traffic and break its privacy guarantees, as shown in Figure 2. Indeed, the Snowden documents revealed that the NSA and GCHQ attempted to break the privacy provided by Tor in this manner (as well as by "contributing" their relays to the system) [24].

Accepted to CACM, 2019, to appear
2019. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Recent research on systems that hide metadata focuses on dealing with a global adversary with the ability to monitor and inject network traffic on any link, comparable to nation-state organizations like the NSA or GCHQ. For example, various recent systems hide metadata for point-to-point messaging [1, 19, 27, 35, 36], and others facilitate anonymous postings on public bulletin boards [3, 18] in the presence of such an adversary.

Hiding metadata is complicated, and comes at a price. Metadata includes crucial information for functionality. As one notable example, the source and destination addresses, which identify the communication end-points, are included in every IP packet and are fundamental for establishing communication over the Internet. Other than explicit information recorded in network packets, a metadata private communication system also needs to obscure traffic correlations, such as the relation between the time at which messages were sent and the time they were delivered. If the attacker sees that Bob starts receiving messages when Alice starts sending messages and stops receiving messages when she stops sending them, then he can deduce that they are communicating even if the source and destination addresses in messages are hidden. Worse yet, the attacker might control Alice's link to the Internet which allows him to perturbate her connection to trigger such events. For example, the attacker may drop some of Alice's messages and observe correlated reductions in throughput at Bob's end [11, 21]. As a result, many of the connections through Tor are vulnerable to nation-state adversaries [26], and the substantial interest of the intelligence community in metadata [13, 28] means that these attacks are a real problem in practice.

To hide metadata, the communication system needs to change the fundamentals that facilitate efficient communication over the Internet, such as packet routing and timely message delivery. The system might also need to constantly send messages, to hide when the user is actually communicating. These changes lead to significant challenges in designing practical metadata hiding communication systems and result in substantial overhead over the non-metadata-hiding "vanilla" counterparts. The overhead deems some applications such as private voice and video chats, and some less-powerful devices like mobile phones, unusable with the current state of the art.

In this article, we discuss the challenges ahead in hiding metadata to facilitate private communication through common types of applications. That is, allow two users who know one another to communicate without exposing that they are doing so. Hiding communication metadata can also facilitate anonymous communication, where users access a network service without revealing information about their identity. Some services refuse or throttle connections from anonymous users (e.g., connected through Tor) [17, 29], which dispirits adoption. We focus the discussion on the simpler problem of facilitating private communication at large scale between users who want to communicate with one another. Informally, we ask: *is it possible to hide metadata for popular types of applications for user-to-user communication?* We show that some compromises in security and privacy must be made to support large-scale latency-sensitive applications. We motivate why, despite these compromises, supporting such applications with substantial privacy and security guarantees may be possible even with a large user-base.

2 DEALING WITH NATION-STATE ADVERSARIES

Designing systems that hide metadata in the face of resourceful organizations like nation-states requires dealing with several attack vectors: A nation-state might monitor traffic on network links, not just within its territory, but also at a global scale. It might also corrupt a substantial fraction of the system's servers in targeted attacks to read messages encrypted for these servers and find correlations between messages that they relay. Moreover, attacks are not confined to be technical. Nation-states might compel service providers to cooperate with them. For example, the NSA's PRISM program coerced public companies to disclose data that was not otherwise observable by the NSA [12].

To understand where the performance penalties in dealing with these challenges stem from, we explain in more detail the attacks that metadata private communication systems need to resist and the defenses that these systems typically deploy.

Aside from the challenges listed above, which aim to break privacy, nation-states may also target the availability of a metadata-hiding service. They may block the service to stop users from communicating privately or even detain users who install the system's clients. We discuss why the key to dealing with such problems may lie in achieving good performance.

2.1 Traffic monitoring

IP addresses, service ports, message sizes and their transmission/receipt times are all directly observable to an attacker monitoring the traffic links. These observations reveal information about the underlying conversation, as discussed earlier. Traffic monitoring attacks are often also tough to detect since the attacker can remain passive, i.e., avoid intervening with actual communication.

Internet routing increases the attack surface. Messages travel on the Internet through many hops and via several autonomous systems (independently administrated organizations). It is sufficient for an attacker to tap any of these links, or coerce any of the transient autonomous systems, to observe communication metadata. Internet routing gives end-users very little control over the routes their messages take. They can choose their Internet service provider, but that provider selects the next hop (and that hop selects the next one). So forcing their messages to avoid routes through particular organizations is difficult. Worse yet, Internet routing is asymmetric, so messages traveling from Alice to Bob are likely to go over different links than those from Bob to Alice. Most protocols rely on bi-directional communication, which essentially doubles the attack surface [31]. See illustration in Figure 3. In particular, any protocol that relies on TCP (the Internet's transmission control protocol) implicitly requires recipients to acknowledge every message that they get, which makes 40% of connections through Tor vulnerable to nation states that passively tap the network links in their territory [26].

The underlying protocols that facilitate communication over the Internet were not designed to be secure. As a result, even if

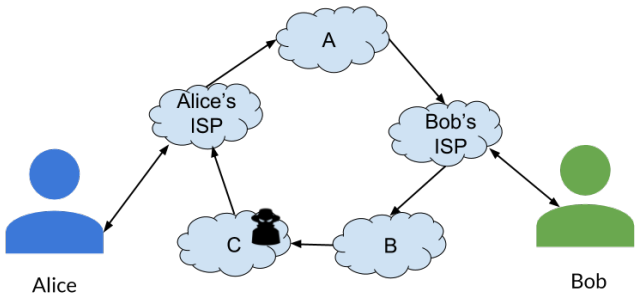

Figure 3: The attacker can only observe traffic going through autonomous system C, this allows him to monitor traffic from Bob to Alice and learn that they are communicating.

traffic would not typically route through an attacker-controlled organization, changing the traffic's path is usually feasible and with minimal effort. Mounting prefix hijacks only requires administrating one autonomous system and announcing someone else's address space. The false announcement manipulates the Internet's routing protocol into delivering to the attacker messages intended to the (hijacked) address space. In fact, Internet routing is so fragile that it is tough to distinguish between common configuration mistakes and deliberate attacks, providing attackers a reasonable excuse even if they are detected. In 2013, an ISP in Belarus exploited Internet routing to hijack Internet traffic intended for Iceland [4]. In a similar, but believed to be accidental, incident in 2014 Indosat hijacked traffic for 44 Tor relays (as well as other destinations) [40].

Dealing with traffic monitoring. To mitigate traffic monitoring attacks, it is crucial to ensure that traffic patterns appear the same no matter which users communicate and regardless of the time, length, and content of their conversation. To this aim, most systems that hide metadata route fixed-size messages through a relay server, which unlinks input messages from output messages.

The relay server shuffles the output messages' transmission order. Since the shuffle permutation is secret, the adversary cannot map a message output from the relay back to the corresponding input. (Hop-to-hop encryption between the users and the relay unlinks the contents of the relay's inputs from its outputs.) In this manner, directly observable fields like source and destination addresses do not reveal information about the user-to-user communication metadata. To prevent timing correlations, most systems are synchronous. They operate in rounds, where at the beginning of the round each user submits a message, and at the end of the round, the system delivers the message to its destination. Synchronicity allows dealing with timing attacks since all message exchanges happen on round boundaries.

Routing through a relay often forces messages through unduly long routes, and synchronicity implies that a relay cannot forward even a single message before it is done processing the entire batch of messages for a particular round. Another unfortunate consequence of synchronous designs is that they require all users to keep submitting messages to the system at every round or it becomes apparent when users are involved in a conversation. Short rounds would facilitate low latency communication, but require everyone to send messages all of the time. Having clients constantly send messages increases the load on the system and the cost for the clients. It implies that operating a client has high network and energy costs that make existing synchronous solutions prohibitively expensive to run on mobile devices, which are limited by data plans and battery life. Support for mobile devices in a synchronous system remains a largely unsolved challenge which we discuss in §4.2.

Indeed, asynchronous systems are more perfomant than others [7, 27] and better accommodate latency-sensitive applications and mobile clients, but may suffer from statistical attacks which correlate users' traffic patterns [7, §3.1],[27, §4.1].

2.2 Corrupt servers and colluding operators

Attackers may compromise the system's servers. Moreover, nation-states might be in a position to coerce a server operator to collude with them [12]. Such attacks would expose the server's secrets to the attacker. In the typical relay-based operation sketched above, the server's secrets would allow the attacker to learn the message-shuffle permutation and map messages sent from the relay server back to its inputs, thereby unveiling the source and destination of each message. One standard approach for resisting malicious servers is to route each message through several relays, each administered by a different organization, as shown for Tor in Figure 1. A typical design goal is to guarantee that if any of these servers are honest, metadata remains hidden. Of course, processing a message by multiple servers induces latency, adding to the challenge of supporting latency-sensitive applications in practice. Notably, it is possible to avoid trusting any of the system's servers using sophisticated cryptographic constructs such as private information retrieval, which obviates routing messages through multiple relays. We discuss these alternatives in §3.

2.3 Service blocking

An attacker might give-up on breaking the system's privacy guarantees, and block connections to the system altogether or detain its users. In particular, there is evidence that some governments fingerprint and block connections to Tor [2, 38]. One approach for protecting against fingerprinting is to disguise communication as other popular services, which are already perceived legitimate [23, 33, 37]. However, disguising the service requires keeping its server addresses hidden to avoid blacklisting (see discussion in [23]). The approach taken by the Tor project to protect against blacklisting is to secretly distribute ephemeral addresses of "bridge" nodes, relay servers to which users directly connect to access Tor. Bridges ultimately get discovered and blocked, creating a cat and mouse game between Tor's operators and some governments.

A perhaps more concerning problem is that attackers might consider running the system's client to be suspicious in its own right. It seems that the most effective way to combat this concern is to make the metadata-hiding service so popular, such that using it does not raise suspicion (an argument originally made by Tor's designers [6]). Making a privacy-preserving service appeal to a broad audience of users, who often do not worry much about their privacy, requires achieving comparable performance to the available non-metadata-hiding alternative. Achieving good performance together with the design constraints forced by the privacy

or security requirements is difficult. The next section focuses the discussion on this crucial goal.

3 ONLINE PRIVACY AT SCALE

A large user-base is crucial for privacy, so metadata-hiding communication systems need to be designed to scale. One standard approach to scaling a system, in general, is to design it such that the more servers are available, the more users it can support. In the context of metadata-hiding systems, which typically rely on volunteer organizations to contribute and operate servers, we would like the performance to improve as the number of those organizations grows. This property is known as *horizontal scaling*. (Contrasted to vertical scaling, which requires every provider to contribute beefier machines and more bandwidth.) Recent research shows a fundamental tradeoff [5]: to achieve low-latency communication without compromising on privacy, the system must increase its bandwidth proportionally to the number of users; so horizontal scaling should be treated as a first-class design principle, as it allows to keep the load on each contributing organization moderate even when the user-base grows large. Through horizontal scaling, Tor can serve millions of concurrent users. It distributes the client-load across more than 6,000 servers and reaches acceptable latency to support a variety of Internet applications.[1]

The research community's efforts to build metadata-private systems that can resist nation-state adversaries have so far resulted in systems that provide medium to high latency, ranging from several seconds to hours. Three recent examples, Atom, Stadium and Karaoke [18, 19, 35], have shown how to design such communication systems that scale horizontally. Notably, with 100 organizations contributing servers, Karaoke can exchange messages between 2 million users every 7 seconds — over 21 billion messages per day [19]. Exchanging this many messages per day falls in the ballpark of popular (encrypted, but not metadata-hiding) messaging applications such as WhatsApp and Telegram, which reported delivering 55 billion and 15 billion messages per day [32, 34]. However, relatively high latency has limited the applications for metadata-private systems. For example, it may be acceptable for an e-mail to reach the recipient's mailbox with a latency of a few minutes, but messaging applications are expected to deliver within seconds, and voice and video chats require sub-second latency for adequate user experience which is key for broad adoption. This limitation is unfortunate since latency-sensitive communication mediums, such as VoIP calls and video conferencing, are popular. For example, in 2013, Microsoft reported 2 billion minutes of Skype calls every day [22].

The primary challenge in designing communication systems that hide metadata is achieving a combination of three crucial goals: providing strong privacy guarantees, resisting attacks from nation-state adversaries, and attaining sufficient performance to support a target higher-level application (such as e-mails, text messaging, or voice chats) with the adequate user experience.

3.1 The cost of perfect guarantees

As we next discuss, targeting the best privacy and security guarantees leads to performance problems. Specifically, challenging the ability of the design to scale to support a large user-base.

Privacy. It is possible to design systems that avoid leaking any information at the cost of excessive communication or computation. For example, a system where each user sends every message to all others can resist passive and active attacks [3]. Cryptographic techniques like DC-nets [39] and Private Information Retrieval [1], keep the client's communication cost to a small constant, but introduce computational overheads that are quadratic in the number of users. These designs [1, 3, 39] provide the strongest form of privacy that users could hope for, but due to the overhead of these schemes performance suffers severely as the user-base grows.

However, even if the system facilitates communication without leaking any information about metadata, some information might leak just by the limitations of a human user. For example, if Alice is currently in a VoIP conversation with Bob, she is probably incapable of simultaneously talking with another user. Therefore, if the attacker tries to call Alice and Bob at the same time and the two do not respond, then he learns some statistical information about the possibility that they are communicating. By repeating this experiment, the attacker might find that Alice and Bob's availability is correlated, and reach an informed conclusion about whether they communicate. A fundamental question in designing metadata-hiding systems is therefore what kind of leakage is acceptable for the particular application, and whether we can trade some leakage to get better performance.

Strength of trust assumptions. Systems must have clear underlying trust assumptions to provide meaningful privacy guarantees. Weaker assumptions mean that the system's privacy guarantees are more likely to hold in practice. The weakest form of trust assumption, referred to as zero-trust, is not trusting the system at all. It means that the system's servers facilitate communication, but even if they all turn out to be malicious, the system maintains its privacy guarantee — to keep communication metadata hidden (although malicious servers might still prevent users from communicating).

The zero trust assumption inevitably comes with a significant computational cost. To understand why, consider an idealized scenario where users deposit messages at a server, and each user can poll that server for messages intended for them; fetching a message without leaking information about which one is being fetched can be done using cryptographic protocols for private information retrieval. Fundamentally, however, the server must process all deposited messages for each user query to be untrusted. Otherwise, the server must know that some user could not have sent a message to some other user, so some information about the metadata was surely exposed to the server. This implicit computational requirement leads to significant challenges in supporting many users. As one concrete example, Pung [1] uses zero-trust as its underlying security assumption, but with 2M users, the communication latency grows to 18 minutes.

[1] See https://metrics.torproject.org for measurements about Tor's deployment.

3.2 The 3-way tradeoff

To get around the performance challenges in §3.1 it seems necessary to compromise on weaker privacy guarantees and stronger trust assumptions. We next describe possible compromises and what they enable to achieve in performance. Table 1 summarizes the tradeoff points of several recent proposals.

Privacy *vs.* performance. Hiding all communication metadata, no matter what actions the attacker takes, results in significant overhead. However, as a recent design shows, it is possible to efficiently prevent any leakage of information about a conversation's metadata when the attacker is passive [19]. This is important because, as we noted earlier, passive attacks are tough to detect.

It is also possible to avoid severe performance penalties in case the attacker is active by relaxing the privacy goal to allow leaking a small amount of statistical information on every message exchange. Systematic mechanisms for quantifying and limiting information leakage to an adversary were studied through differential privacy [8]. Differentially private systems protect an individual's data by stochastically noising the system's outputs which the attacker can observe. These systems provide a weaker notion of privacy than those that rely on message broadcast or private information retrieval (§3.1), but have seen significant adoption in practice (e.g., by Google [9] and Apple [14]).

In context of communication systems, differential privacy limits information leakage by adding dummy messages as cover traffic [19, 20, 35, 36]. The system's servers generate these dummies, and their amount is decided at random by each server in every communication round according to a fixed distribution (set by the system's designers). The stochastic process of adding dummy messages to user messages limits what the attacker could learn on the actual user-to-user communication from whatever attack he executes. The more dummy messages that the system processes, the less information that might leak on each message-exchange. The performance benefit of differential privacy stems from the fact that the quantity of dummies is independent of the number of users and messages they exchange.

The attacker might carry active attacks over many message exchanges to entice sufficient leakage of information about the communicating pairs of users. To prevent the attacker from aggregating this information into something useful, a system can leverage the fact that active attacks may be detected. When an active attack is detected, a security policy could stop sensitive communication for some time or until the user connects through a different network (where the attacker might not be present). Such a policy would limit the attacker's ability to leak significant information about sensitive metadata. Importantly, clients must continue submitting messages to the system even after disconnecting from their peers, to avoid leaking that they were involved in a conversation.

Strength of trust assumptions *vs.* performance. Zero trust induces quadratic costs in the number of users, which limits the ability to support a large user-base (§3.1). It is therefore important to identify weaker trust assumptions that are likely to hold in practice. Two other forms of trust assumptions are common.

Any-trust. One common alternative is to deploy the communication service over several servers administered by independent organizations, and to trust that at least one of these organizations is honest (without requiring users to trust a specific one). Distributing trust across many servers in the context of metadata-private communication systems dates back to Chaum's mixnets in the 80s. By relaxing the security goal from zero-trust to any-trust, we can avoid the overhead of scanning all senders' inputs for each output that the system provides to a recipient. In practice, Vuvuzela [36], the most performant any-trust system, can exchange messages between 2M users in about a minute (over an order of magnitude of improvement in latency and throughput over Pung).

A minute of latency is, however, far from being on par with vanilla messaging applications. A key reason for this performance handicap is that the any-trust assumption excludes horizontal scaling. Since under this assumption there may be just one honest server in the entire system, each server must process all messages (otherwise some messages might have skipped processing by the honest server, leaving their metadata exposed). In contrast, vanilla applications distribute the load over many servers.

A fraction of the system's servers are honest. It seems inevitable to require a stronger trust assumption than zero- or any-trust to be able to scale the system to support a large user-base. A much more performance-friendly assumption is that some fraction of the system's servers are honest. It allows sharding the load of user messages among servers (enabling horizontal scaling) as shown in [18, 19, 35]. Under this assumption, a user's message may be processed only by a small subset of the servers, where one server in the subset is assumed to be honest (i.e., each subgroup of servers is any-trust). Karaoke, which operates under this assumption, improved on Vuvuzela's performance by about an order of magnitude in latency and throughput [19].

4 DISCUSSION

We have so far discussed the challenges in building popular means of communication that hide metadata. Progress in research on metadata-hiding systems is promising. We next motivate why building such systems that support low-latency applications like voice and video chats, may be tangible. We then discuss the remaining challenges that the research community would need to tackle to make it happen.

4.1 Metadata-private low-latency communication at scale is tangible

Let us start by some reasons for optimism, motivating why providing meaningful metadata privacy guarantees for many types of applications may be feasible.

Focusing on human communication. The volume of machine-to-machine communication proliferates, and constant improvements in network infrastructure support this growth. However, latency-sensitive human-to-human communication typically involves lively interactions between people and as such does not grow as rapidly as machine-to-machine communication (as one example, the average monthly mobile voice minutes per person in the UK increased only by 10% between 2008 to 2013 [30]). The

better performance ↑

System	privacy	trust assumption	latency (at 2M clients)	throughput	horizontally scalable?
Tor [7]	vulnerable	one out of client-selected servers is honest	sub-second	very high (millions of users' web traffic)	✓
Karaoke [19]	differential privacy	80% honest	7 sec	571k msgs/sec	✓
Vuvuzela [36]	differential privacy	any-trust	1 min	50k msgs/sec	X
Pung [1]	no metadata leak	zero-trust	18 min	2k msgs/sec	✓

↓ better privacy & security

Table 1: The 3-way tradeoff in state of art systems. More performant systems provide weaker privacy and security guarantees. (Karaoke and Pung are horizontally scalable and evaluated with 100 servers.)

difference in growth rates suggests that systems' capabilities are catching up with the amount of human-generated communication, and its metadata that we might wish to hide.

Leaking some information may be acceptable. As we discussed in §3, even a perfect metadata-hiding communication system cannot prevent the attacker from learning something about the communication. The challenge, therefore, lies in leaking as little information as possible while providing solid performance. Differential privacy seems to be a promising direction in consolidating this tradeoff. It allows to limit and quantify the amount of information the system leaks to an attacker and allows to circumvent the computation and communication overheads of other approaches (where the system leaks no additional information about its users' traffic).

4.2 Challenges ahead

Recent work shows that metadata-private systems can provide meaningful privacy guarantees and support a large user-base. In the last eight years, the throughput of metadata-private systems has increased from about 420K messages per hour [39] to over a billion messages per hour [19]. Achieving this improvement is accredited to theoretical advances in cryptographic constructs and differential privacy, horizontally scalable designs, and engineering efforts. Yet the current state-of-the-art still induces significant performance penalties. We identify three remaining challenges that current systems face, which, if alleviated, could dramatically improve performance and drive user adoption.

Supporting mobile devices. A common approach to avoid exposing correlations between the times that Alice and Bob are online and might be communicating is to have the clients regularly send messages (sending dummies if she is not involved in a conversation). This approach, which was taken by state of the art systems, induces significant cost to battery life and data-plans when running on mobiles (§2.1). To support mobile clients, which are used by a massive portion of the Internet's users, it seems necessary to rethink this strategy.

It might be possible to tackle this problem by weakening the privacy guarantees. One direction is to support two types of users: (A) those who are always online, and (B) those who are online only when they communicate. It seems feasible to protect metadata when users from both groups chat with one another. To motivate why, consider the simplest scenario where Bob is the only user at group B, i.e., the only one who is online only when he communicates. Then an adversary observing the network traffic sees when Bob is communicating, but cannot tell whether Bob is chatting with Alice or any of the other users at group A (who are always online). In other words, the system would reveal that Bob is chatting with some user (and when) but would hide that this user is Alice.

Reducing computations. The recent horizontally scalable designs distribute the communication overhead over many contributing organizations. The state of the art systems, however, make extensive use of public key cryptography and rely on hefty cryptographic protocols like zero-knowledge proofs of shuffle and correct decryption. In particular, the performance of the horizontally scalable systems in Table 1, Karaoke and Pung, is bounded by computations (see experimental evaluation [1, 19]). Finding a way to minimize the use of these cryptographic constructs, such as by establishing persistent private sessions and using symmetric-key cryptography within these sessions (as often done when hiding content), would alleviate the computational bottleneck and reduce communication latency significantly.

Improving the topology. It is common to route messages through servers operated by organizations in different political and geographic regions, to reduce the chance that the organizations administrating these servers would collude. Topology studies were performed mostly on Tor, to avoid routing through specific autonomous systems [16, 25] (combating the attacks in §2) and to avoid overloading specific relays [15].

A largely remaining challenge is to optimize the route that messages would take through different geographic regions, so as to avoid sending messages through an overly large distance. A route with many distant randomly-selected hops (as in [18, 19, 35, 36]) means that even if each server only relays a small portion of the messages, and does not perform any computationally-heavy processing, the aggregate of the inter-server latency might be too expensive to support some applications. A major challenge in supporting latency-sensitive applications is therefore to identify better routing topologies, which allow to mix all messages for privacy, yet do not require messages to go through many hops for performance.

REFERENCES

[1] Sebastian Angel and Srinath T. V. Setty. 2016. Unobservable Communication over Fully Untrusted Infrastructure. In *OSDI*, Kimberly Keeton and Timothy Roscoe (Eds.). USENIX Association, 551–569.

[2] Simurgh Aryan, Homa Aryan, and J. Alex Halderman. 2013. Internet Censorship in Iran: A First Look. In *FOCI*, Jedidiah R. Crandall and Joss Wright (Eds.). USENIX

Association.

[3] Henry Corrigan-Gibbs, Dan Boneh, and David Mazières. 2015. Riposte: An Anonymous Messaging System Handling Millions of Users. In *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 321–338. http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7160813

[4] Jim Cowie. 2013. New Threat: Targeted Internet Traffic Misdirection. http://www.renesys.com/2013/11/mitm-internet-hijacking. (2013).

[5] Debajyoti Das, Sebastian Meiser, Esfandiar Mohammadi, and Aniket Kate. 2018. Anonymity Trilemma: Strong Anonymity, Low Bandwidth, Low Latency — Choose Two. In *Security and Privacy*. IEEE.

[6] Roger Dingledine and Nick Mathewson. 2006. Anonymity Loves Company: Usability and the Network Effect. In *Workshop on the Economics of Information Security*.

[7] Roger Dingledine, Nick Mathewson, and Paul F. Syverson. 2004. Tor: The Second-Generation Onion Router. In *USENIX Security Symposium*, Matt Blaze (Ed.). USENIX, 303–320.

[8] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *TCC*, Vol. 3876. Springer, 265–284.

[9] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *ACM Conference on Computer and Communications Security*, Gail-Joon Ahn, Moti Yung, and Ninghui Li (Eds.). ACM, 1054–1067. http://dl.acm.org/citation.cfm?id=2660267

[10] Klint Finley. 2017. Half of the Internet is now encrypted. This makes everyone safer. https://www.wired.com/2017/01/half-web-now-encrypted-makes-everyone-safer/. (2017).

[11] Yossi Gilad and Amir Herzberg. 2012. Spying in the Dark: TCP and Tor Traffic Analysis. In *Privacy Enhancing Technologies (LNCS)*, Simone Fischer-Hübner and Matthew K. Wright (Eds.), Vol. 7384. Springer, 100–119.

[12] Glenn Greenwald and Ewen MacAskill. 2013. NSA Prism program taps in to user data of Apple, Google and others. https://www.theguardian.com/world/2013/jun/06/us-tech-giants-nsa-data. (2013).

[13] Michael Hayden. 2014. The Price of Privacy: Re-Evaluating the NSA. Johns Hopkins Foreign Affairs Symposium. (April 2014). https://www.youtube.com/watch?v=kV2HDM86XgI&t=17m50s.

[14] Apple inc. 2016. Differential Privacy. https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf. (2016).

[15] Aaron Johnson, Rob Jansen, Nicholas Hopper, Aaron Segal, and Paul Syverson. 2017. PeerFlow: Secure Load Balancing in Tor. *PoPETs* 2017, 2 (2017), 74–94.

[16] Aaron Johnson, Chris Wacek, Rob Jansen, Micah Sherr, and Paul F. Syverson. 2013. Users get routed: Traffic correlation on Tor by realistic adversaries. In *ACM Conference on Computer and Communications Security*, Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung (Eds.). ACM, 337–348. http://dl.acm.org/citation.cfm?id=2508859; http://dl.acm.org/citation.cfm?id=2541806

[17] Sheharbano Khattak, David Fifield, Sadia Afroz, Mobin Javed, Srikanth Sundaresan, Damon McCoy, Vern Paxson, and Steven J. Murdoch. 2016. Do You See What I See? Differential Treatment of Anonymous Users. In *NDSS*. The Internet Society. http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2017/09/do-you-see-what-i-see-differential-treatment-anonymous-users.pdf

[18] Albert Kwon, Henry Corrigan-Gibbs, Srinivas Devadas, and Bryan Ford. 2017. Atom: Horizontally Scaling Strong Anonymity. In *SOSP*. ACM, 406–422. http://dl.acm.org/citation.cfm?id=3132747

[19] David Lazar, Yossi Gilad, and Nickolai Zeldovich. 2018. Karaoke: Fast and Strong Metadata Privacy with Low Noise. In *OSDI*. USENIX Association.

[20] David Lazar and Nickolai Zeldovich. 2016. Alpenhorn: Bootstrapping Secure Communication without Leaking Metadata. In *OSDI*, Kimberly Keeton and Timothy Roscoe (Eds.). USENIX Association, 571–586.

[21] Brian Neil Levine, Michael K. Reiter, Chenxi Wang, and Matthew K. Wright. 2004. Timing Attacks in Low-Latency Mix Systems (Extended Abstract). In *Financial Cryptography (Lecture Notes in Computer Science)*, Ari Juels (Ed.), Vol. 3110. Springer, 251–265.

[22] Microsoft. 2013. 2 Billion Minutes a Day! Skype blog. https://blogs.skype.com/stories/2013/04/03/thanks-for-making-skype-a-part-of-your-daily-lives-2-billion-minutes-a-day/. (2013).

[23] Hooman Mohajeri Moghaddam, Baiyu Li, Mohammad Derakhshani, and Ian Goldberg. 2012. SkypeMorph: Protocol obfuscation for Tor bridges. In *ACM Conference on Computer and Communications Security*, Ting Yu, George Danezis, and Virgil D. Gligor (Eds.). ACM, 97–108. http://dl.acm.org/citation.cfm?id=2382196

[24] National Security Agency. 2013. Tor Stinks. The Guardian. (Oct. 2013). https://www.theguardian.com/world/interactive/2013/oct/04/tor-stinks-nsa-presentation-document.

[25] Rishab Nithyanand, Rachee Singh, Shinyoung Cho, and Phillipa Gill. 2016. Holding all the ASes: Identifying and Circumventing the Pitfalls of AS-aware Tor Client Design. *CoRR* abs/1605.03596 (2016). http://arxiv.org/abs/1605.03596

[26] Rishab Nithyanand, Oleksii Starov, Phillipa Gill, Adva Zair, and Michael Schapira. 2016. Measuring and Mitigating AS-level Adversaries Against Tor. In *NDSS*. The Internet Society. http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2017/09/measuring-mitigating-as-level-adversaries-against-tor.pdf

[27] Ania M. Piotrowska, Jamie Hayes, Tariq Elahi, Sebastian Meiser, and George Danezis. 2017. The Loopix Anonymity System. In *USENIX Security Symposium*, Engin Kirda and Thomas Ristenpart (Eds.). USENIX Association, 1199–1216.

[28] Alan Rusbridger. 2013. The Snowden Leaks and the Public. http://www.nybooks.com/articles/2013/11/21/snowden-leaks-and-public/, organization=The New York review of books. (2013).

[29] Rachee Singh, Rishab Nithyanand, Sadia Afroz, Paul Pearce, Michael Carl Tschantz, Phillipa Gill, and Vern Paxson. 2017. Characterizing the Nature and Dynamics of Tor Exit Blocking. In *USENIX Security Symposium*, Engin Kirda and Thomas Ristenpart (Eds.). USENIX Association, 325–341.

[30] Statistica, the statistics portal. 2013. Average monthly outbound mobile voice minutes per person in the United Kingdom (UK) from 2008 to 2013 (in minutes). (2013).

[31] Yixin Sun, Anne Edmundson, Laurent Vanbever, Oscar Li, Jennifer Rexford, Mung Chiang, and Prateek Mittal. 2015. RAPTOR: Routing Attacks on Privacy in Tor. In *USENIX Security Symposium*, Jaeyeon Jung and Thorsten Holz (Eds.). USENIX Association, 271–286.

[32] Telegram. 2017. 15 Billion Telegrams Delivered Daily. Telegram announcement, https://telegram.org/blog/15-billion. (2017).

[33] The Tor project. 2017. Pluggable Transports. https://www.torproject.org/docs/pluggable-transports. (2017).

[34] Liam Tung. 2017. WhatsApp: Now one billion people send 55 billion messages per day. http://www.zdnet.com/article/whatsapp-now-one-billion-people-send-55-billion-messages-per-day/. (2017).

[35] Nirvan Tyagi, Yossi Gilad, Derek Leung, Matei Zaharia, and Nickolai Zeldovich. 2017. Stadium: A Distributed Metadata-Private Messaging System. In *SOSP*. ACM, 423–440. http://dl.acm.org/citation.cfm?id=3132747

[36] Jelle van den Hooff, David Lazar, Matei Zaharia, and Nickolai Zeldovich. 2015. Vuvuzela: scalable private messaging resistant to traffic analysis. In *SOSP*, Ethan L. Miller and Steven Hand (Eds.). ACM, 137–152. http://dl.acm.org/citation.cfm?id=2815400

[37] Zachary Weinberg, Jeffrey Wang, Vinod Yegneswaran, Linda Briesemeister, Steven Cheung, Frank Wang, and Dan Boneh. 2012. StegoTorus: a camouflage proxy for the Tor anonymity system. In *ACM Conference on Computer and Communications Security*, Ting Yu, George Danezis, and Virgil D. Gligor (Eds.). ACM, 109–120. http://dl.acm.org/citation.cfm?id=2382196

[38] Philipp Winter and Stefan Lindskog. 2012. How the Great Firewall of China is Blocking Tor. In *FOCI*, Roger Dingledine and Joss Wright (Eds.). USENIX Association.

[39] David Isaac Wolinsky, Henry Corrigan-Gibbs, Bryan Ford, and Aaron Johnson. 2012. Dissent in Numbers: Making Strong Anonymity Scale. In *OSDI*, Chandu Thekkath and Amin Vahdat (Eds.). USENIX Association, 179–182.

[40] Earl Zmijewski. 2013. Indonesia Hijacks the World. https://dyn.com/blog/indonesia-hijacks-world/. (2013).