Understanding Tor Usage with Privacy-Preserving Measurement

Akshaya Mani*
Georgetown University
am3227@georgetown.edu

T Wilson-Brown*
UNSW Canberra Cyber
University of New South Wales
teor@torproject.org

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

Aaron Johnson
U.S. Naval Research Laboratory
aaron.m.johnson@nrl.navy.mil

Micah Sherr
Georgetown University
msherr@cs.georgetown.edu

ABSTRACT

The Tor anonymity network is difficult to measure because, if not done carefully, measurements could risk the privacy (and potentially the safety) of the network's users. Recent work has proposed the use of differential privacy and secure aggregation techniques to *safely* measure Tor, and preliminary proof-of-concept prototype tools have been developed in order to demonstrate the utility of these techniques. In this work, we significantly enhance two such tools—PrivCount and Private Set-Union Cardinality—in order to support the safe exploration of new types of Tor usage behavior that have never before been measured. Using the enhanced tools, we conduct a detailed measurement study of Tor covering three major aspects of Tor usage: how many users connect to Tor and from where do they connect, with which destinations do users most frequently communicate, and how many onion services exist and how are they used. Our findings include that Tor has ~8 million daily users, a factor of four more than previously believed. We also find that ~40% of the sites accessed over Tor have a torproject.org domain name, ~10% of the sites have an amazon.com domain name, and ~80% of the sites have a domain name that is included in the Alexa top 1 million sites list. Finally, we find that ~90% of lookups for onion addresses are invalid, and more than 90% of attempted connections to onion services fail.

ACM Reference Format:
Akshaya Mani, T Wilson-Brown, Rob Jansen, Aaron Johnson, and Micah Sherr. 2018. Understanding Tor Usage with Privacy-Preserving Measurement. In *2018 Internet Measurement Conference (IMC '18), October 31-November 2, 2018, Boston, MA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3278532.3278549

1 INTRODUCTION

The Tor network has been in operation since 2003 [17], enabling millions of daily users [43] to anonymously access the Internet. Despite its relative longevity, growing popularity, and prominence as a privacy-enhancing communication tool, surprisingly little is known about *who* uses Tor and *how* they use it.

*Equally credited authors.

IMC '18, October 31-November 2, 2018, Boston, MA, USA

ACM ISBN 978-1-4503-5619-0/18/10…$15.00
https://doi.org/10.1145/3278532.3278549

Challenges in Measuring Tor. The primary challenge in conducting measurements on the Tor network is that, if not done with extreme care, they can pose significant risk to the network's users. The *collection* of statistical information implies the need to store data about clients, traffic patterns, sites visited, *etc.*—information that may otherwise not be collected at the relays for privacy reasons. Even if the entities collecting this information are honest, the exposure of the data (*e.g.*, through machine compromise or compulsion from government authorities) could pose serious privacy risks. In addition, malicious relay operators may combine their local observations with the global information from published statistics to deanonymize some network users. For instance, an entry relay might combine published information about sites recently visited with the client IPs that have recently connected to the relay to determine those clients' destinations. Furthermore, public aggregated measurement data risks users' privacy since adversaries could potentially combine this data with other background knowledge (*e.g.*, when a tweet was posted, ISP logs showing user accesses to the Tor network) to deanonymize users [15].

In addition to user privacy and safety, monitoring communication is antithetical to the Tor Project's mission. For this reason, the Tor Metrics Portal [43], a data repository that archives information about the anonymity network's size, makeup, and capacity, provides only a limited number of statistics, many of which are based on indirect measurements and assumptions of Tor client behavior. We demonstrate in this paper that some of Tor's estimates do not correlate well with our own direct observations of user behavior.

Other efforts to measure Tor, such as those that record packets at Tor's ingress and egress points, can pose serious ethical and legal issues: they rely on unsafe [41] (and potentially unlawful [40]) techniques that would be difficult to repeat. As a result, such studies have been met with criticism by the privacy community [40].

Toward Safe Tor Measurement. There has been an exciting recent trend in the literature that proposes differentially private [20] statistics collection techniques for anonymity networks such as Tor. Systems such as PrivEx [22], PrivCount [26], HisTorϵ [35], and PSC [23] enable useful statistics collection while providing formal guarantees about the privacy risks they impose on the network's users. However, to date, researchers have (rightfully) focused on developing these privacy-preserving measurement techniques rather than on performing exhaustive measurements of deployed anonymity networks such as Tor.

Our Contributions. This paper presents the largest and most comprehensive measurement study of the Tor network to date, performed using newly proposed differentially private statistics collection techniques. To conduct our study, we modify Tor and significantly enhance and improve PrivCount [26] and PSC [23] (described in more detail in the following section) to support a larger variety of measurements, and contribute our improvements to the respective open source projects. We run 16 Tor relays that contribute approximately 3.5% of Tor's bandwidth capacity, and deploy our measurement systems across these relays in order to safely collect measurements of the live Tor network. We extend previous statistical methodology to enable unique-counting on Tor, and extend privacy definitions ("action bounds") to cover new types of user activity. We also describe the challenges and tactics for selecting appropriate privacy parameters to ensure Tor users' safety, as well as methods for inferring whole-network statistics given local observations at our relays.

Using our improved measurement tools and extended techniques, we conduct a detailed measurement study of Tor covering three major aspects of Tor usage: who connects to Tor and from where do they connect; how is Tor used and with what services and destinations do Tor users communicate; and how many onion services exist and how are they used. Findings from our client-based measurements suggest that Tor has ~8 million daily users, which is a factor of four more than previously believed. We are the first to measure client churn in Tor, and as expected we find that the client churn rate decreases over time. We also find that ~40% of the sites accessed over Tor have a torproject.org domain name, ~10% of the sites have an amazon.com domain name, and ~80% of the sites have a domain name that is included in the Alexa top 1 million sites list. Finally, we find that ~90% of onion address lookups fail because the address is missing or the request is malformed, and more than 90% of attempted connections to onion services fail because the server never completes its side of the connection protocol. Our findings both reinforce existing beliefs about the Tor network (*e.g.*, that the vast majority of its use is for web browsing) and elucidate many aspects of the Tor network that have previously not been explored. In some instances, our measurements—which are based on *direct* observations of behavior on Tor—suggest that existing heuristically-driven estimates of Tor's usage (including the number of Tor users) are highly inaccurate.

Limiting the risk to Tor's users was paramount in both the design and implementation of all of our measurements, and a significant contribution of this work is the methodology for choosing system parameters to guarantee adequate levels of protection. We discuss the precautions we took to ensure user safety, argue for the ethical validity of our study, and describe our experiences with institutional ethics boards and an independent safety review board.

2 BACKGROUND

We begin by presenting a brief overview of Tor and reviewing the privacy definitions and privacy-preserving measurement tools we use in our study.

2.1 Tor

Tor provides anonymous TCP connections through source-routed paths that originate at the Tor client (*i.e.*, the user) and traverse through (usually) three relays. Traffic enters the Tor network through *guard relays*. Middle relays carry traffic from guard relays to *exit relays*, where the traffic exits the anonymity network. These anonymous paths through Tor relays are called *circuits*. The unit of transport in Tor circuits is the *cell*: Tor cells carry encrypted routing information and 498 bytes of data [16]. The encryption of application-level headers and payloads makes it difficult for an adversary to perform traffic analysis on circuits and learn the endpoints or content of communication.

Tor has a three-tiered communication architecture. A single circuit may carry multiple *streams*. A Tor stream is a logical connection between the client and a single destination (*e.g.*, a webserver), and is roughly analogous to a standard TCP connection. A request to retrieve a webpage may produce several streams (*e.g.*, to example.com and ads.example.com) that are multiplexed over a single circuit. Finally, Tor maintains longstanding TLS-encrypted *connections* between relays that are adjacent on some Tor circuit. Communication belonging to different circuits that share a common hop between two relays are multiplexed through a single connection. Similarly, a user's Tor client maintains a single connection to each of its guards, through which multiple circuits (and streams) may be formed.

Tor clients periodically download "directory updates" that describe the consensus view of the network. Clients send all user data through one guard by default, but directory updates are obtained through three guards by default. Tor clients avoiding censorship may connect to the network through *bridges* [37], which are guards whose identities are not public and are disseminated by Tor to users requesting them.

Onion Services. Tor allows services to hide their locations through the use of *onion services* [42]. As a special (but common) case, when the onion service corresponds to a hidden website, we refer to it as an *onionsite*.

To operate an onion service, the operator selects six relays to serve as *introduction points*. It then (1) constructs an *onion service descriptor* that contains its public key and the identities of the chosen introduction points, and (2) forms Tor circuits to each of the introduction points. The descriptor is then stored on six or eight relays [8] (depending on Tor version) using a distributed hash table (DHT) [31]. These relays are called onion service directories (for historical reasons, they are also called "hidden service directories" or "HSDirs"). The DHT is indexed by the descriptor ID, which is derived from the public key. The address of the service is a domain with an .onion suffix, which is derived from the public key.

When a user wishes to access the onion service, it queries the DHT to obtain the service's onion descriptor, which includes the identities of its introduction points. It also chooses a relay to serve as a *rendezvous point* (RP), and constructs a circuit to this RP. Using the public key in the onion descriptor, the user encrypts its choice of the RP and sends it via a Tor circuit to one of the onion service's introduction points. This choice is then forwarded to the onion service, using the existing circuit between the introduction point and the service. Finally, the onion service constructs a circuit to the RP, through which it can communicate with the user.

Importantly, Tor conceals the network locations of both the user and the onion service. The communications between the user and the introduction point, the user and the RP, the onion service and the introduction point, and the onion service and the RP are *all* carried over Tor circuits.

2.2 Differential Privacy

Differential privacy [20] is a privacy definition that provides strong guarantees about how much is revealed by answering database queries. Differential privacy is defined in terms of pairs of *adjacent* databases and guarantees that query responses cannot be used to distinguish such pairs. Typically, databases that differ only in a single user's data are considered adjacent, and then the definition ensures that query responses do not reveal much about any user.

Differential privacy has previously been proposed as a means to safely study the Tor network [22, 23, 26, 35]. In existing work, as in this study, *adjacency* is defined by network activity rather than by user. For example, in its study of visits to censored sites, PrivEx defines adjacency as differing by at most six exit connections per hour [22]. Under this definition of adjacency, differential privacy's guarantees apply to *connections* rather than *users*. We also adopt this model and consider privacy protections over particular *user activities* (for example, visiting onion sites) conducted over a fixed length of time rather than on a per-user basis. The latter is difficult to achieve, since conceptually, a single user (*e.g.*, a botnet) could constitute a significant fraction of the activity on the network.

Differential privacy is typically achieved by adding noise to query results. This noise is added in a controlled manner such that the result of the query is nearly the same for adjacent databases. More formally, a (ϵ, δ)-differentially private mechanism [19] is an algorithm $\mathcal{M}$ such that for all adjacent databases D_1 and D_2 and all $S \subseteq \text{Range}(\mathcal{M})$,

$$\Pr[\mathcal{M}(D_1) \in S] \leq \exp(\epsilon) \times \Pr[\mathcal{M}(D_2) \in S] + \delta$$

where ϵ and δ are parameters that can be used to trade off between privacy and accuracy.

2.3 PrivCount

PrivCount [26] is a distributed measurement system that provides (ϵ, δ)-differentially private statistics about the Tor network.

A PrivCount deployment consists of three components: a tally server (TS), at least one data collector (DC), and at least one share keeper (SK). In an execution of PrivCount, the TS, DCs, and SKs act collectively to report (noisy) counts of the events that were observed during a collection period. PrivCount supports both single number queries (*e.g.*, "how many clients connected during the collection period?") and multiple-counter queries (*e.g.*, "how many Tor connections went to {Google, Amazon, Facebook} during the collection period?") in which the "bins" (i.e., counters) are independent. PrivCount is shown to provide (ϵ, δ)-differential privacy if at least one SK is honest.

We add a new set of events to a patched version of Tor that enables PrivCount and PSC to register for onion-service descriptor upload/download occurrences. We also modify the PrivCount code to securely count the exit, client, and onion service statistics necessary to conduct the experiments described in this paper.

2.4 Private Set-union Cardinality (PSC)

PrivCount provides a safe method of counting observed events across a set of Tor relays. A limitation of PrivCount (and one shared by other privacy-preserving measurement systems [22, 35]) is that it cannot determine the count of *distinct* values among the DCs. For example, while PrivCount can answer queries such as "how many client connections were observed?", it cannot answer "how many *unique* clients connected to Tor?"

To answer questions about the count of distinct values, we use the private set-union cardinality (PSC) protocol introduced by Fenske *et al.* [23]. In PSC, each DC k maintains a set of items $\mathcal{I}_k$. PSC computes the cardinality of the union of those itemsets; i.e., $|\bigcup_k \mathcal{I}_k|$ + noise. PSC achieves (ϵ, δ)-differential privacy and does not expose any element of any itemset to an adversary.

The participants in PSC include one or more DCs and one or more *computation parties* (CPs), the latter of which perform a series of noise additions, verifiable shuffles, re-randomizations, and aggregation to compute the final cardinality (plus noise) of the union. PSC is secure against an active adversary that controls all but one of the CPs. The full scheme is described by Fenske et al. [23].

We modify the implementation of PSC from Fenske et al. [23]. Our first modification is to include a tally server (TS). The TS is untrusted and merely coordinates the execution of the protocol: at the beginning of every measurement period the TS signals the CPs and DCs to start execution, and at the end all CPs and DCs send a *finished* signal to the TS. A malicious TS can collude with a malicious CP or DC and disrupt the flow of the protocol. However, it cannot learn any information from an honest DC as long as at least one CP remains honest. Our second modification enables PSC to accept control-port events from our patched version of Tor, which allows PSC to securely count the number of distinct events observed across a set of relays.

3 METHODOLOGY

Conducting safe measurements on Tor requires careful experimentation design, configuration, and deployment. In what follows, we describe the methodology we employ to collect and analyze measurements of the Tor network.

3.1 Deployment

We use several Tor relays and both PrivCount and PSC to conduct Tor measurements. We first enhanced the PrivCount version of Tor[1] so that the PrivCount events that Tor emits include additional information about connections, circuits, and streams as necessary to answer our research questions. We also add new events that report onion service directory usage information [8, 9]. We ran 16 Tor relays with our enhanced version of Tor (6 exit relays and 10 non-exit relays). The Tor relays were run across 3 different countries—United States, Canada, and France—by 3 operators.

We then enhance PrivCount to support the new information it receives from Tor, and to support several new counter types as necessary for our study. Most significantly, we add support for counting set membership using PrivCount histograms which we use to measure domain distribution (§4), geopolitical distribution in (§5), and onion site distribution in (§6). We also significantly

[1] https://github.com/privcount/tor

extend PSC: we slightly modify the original PSC design to include a TS to coordinate the actions of the DCs and CPs, we engineer PSC to collect the PrivCount events emitted by our relays, and we add support for measuring the number of unique domains (§4), clients (§5), and onion sites (§6). All of our enhancements have been merged into the PrivCount and PSC open-source projects.

We set up a PrivCount deployment containing 1 TS, 3 SKs, and 16 DCs (one for each Tor relay), and we set up a PSC deployment containing 1 TS, 3 CPs, and 16 DCs. We use PrivCount and PSC to repeatedly measure Tor in 24 hour periods, where each period focuses on measuring a small set of statistics. During some measurement periods, we use only the subset of the DCs and relays that are in a position to observe the events of interest in order to reduce operator overhead and reduce the likelihood of failure. Apart from the cases where server(s) was temporarily unavailable, the number of CPs/ SKs we use is greater than or equal to the number of relay operators in all our measurements. To maintain our privacy guarantees, PrivCount and PSC measurements are never conducted in parallel, and we always enforce at least 24 hours of delay between any sequential measurement of distinct statistics.

3.2 Privacy

We protect the privacy of individual Tor users using the methodology developed with PrivCount [26]. This approach applies differential privacy to protect a certain amount of network activity within a certain length of time. Limiting the application of differential privacy to bounded amounts of activity is required for producing accurate results, as otherwise we must protect a hypothetical individual whose activity constitutes the majority of network activity, which would necessarily yield inaccurate results. Fortunately, reasonable Tor activity by an individual should for most actions constitute a small fraction of the total network activity, and so we can provide privacy while providing accurate network measurements.

Our publishing mechanisms formally apply (ϵ, δ) differential privacy to the space of inputs that includes all possible network traces in a given time period. Inputs are considered "adjacent" if they differ only in the activity of a single user within a given length of time, and all differences in that activity stay within a set of *action bounds*. We use 24 hours as the length of time defining adjacency, and our action bounds are shown in Table 1. One way to understand the privacy guarantee that results is that, for any two reasonable sequences of network actions that a user could perform in a 24-hour period, including nothing at all, they most likely cannot be distinguished based on the statistics published by our mechanisms.

We derive the action bounds by considering reasonable activities that a Tor user might perform over 24 hours. In this analysis, we consider protecting the privacy of both Tor clients and onion services. We take into account how certain type of user actions would translate into observable actions on the Tor network, such as creating new circuits or sending data cells, based on our understanding of the Tor protocol and software, taking into account features such as caching and how streams are assigned to circuits. We consider several types of common Tor activities in this analysis: web browsing with Tor Browser, chat with the Ricochet P2P onion service [5], and running a Web server as an onionsite. We compute the amount of network actions that result from reasonable amounts of these

Table 1: Action bounds for measurements.

Action	Daily bound	Defining activity
Connect to domain	20 domains	Web
Send or receive exit data	400 MB	Web
Connect to Tor from new IP address	1 day: 4 IPs 2+ days: 2 IPs	N/A
Create TCP connection to Tor	12 connections	N/A
Create circuit through entry guard	651 circuits	Chat
Send or receive entry data	407 MB	Web
Upload descriptor	450 uploads	Onionsite
Upload descriptor of new onion address	3 addresses	Onionsite
Fetch descriptor	30 fetches	Onionsite
Create rendezvous connection	180 connections	Chat
Send or receive rendezvous data	400 MB	Web or onionsite

activities and use the maximum to define the action bound. The activity that provides the maximum value defining each bound is shown in the final column of Table 1. We release the action bounds as an auxiliary material in the appendix [36].

For example, we choose to protect connecting to 20 domains through an exit circuit, which would protect browsing two new websites for each of 10 hours per day, as the other activities (*i.e.*, chat and onionsites) would not create domain connections. This analysis also allows for additional page loads within the same site, as they would be assigned to the same circuit and thus would not be measured as a new domain connection. Note that certain observable actions apply to all Tor activities, such as creating a TCP connection to a guard, and thus have no defining activity.

We use privacy parameters $\epsilon = 0.3$ and $\delta = 10^{-11}$. The value of ϵ is the same one used by Tor to protect its onion-service statistics [29]. We choose δ such that if there are n Tor users, then δ/n is still small, which ensures that each user is simultaneously protected [21]. For example, if $n = 10^{-6}$ then $\delta/n = 10^{-5}$.

Note that the length of each measurement period only affects the accuracy of the results, as the differentially-private noise ensures privacy for any measurement length. We choose 24-hour periods to yield results that are accurate enough to draw conclusions from (*i.e.*, that are sufficiently large relative to the noise).

3.3 Statistical Analysis

Our measurement techniques do not give us a complete and exact view of the total amount of activity in the Tor network. Because we make measurements from a small subset relays, we only observe a sample of Tor's overall activity. In addition, our privacy-preserving techniques intentionally include some random errors in the measurement. We use statistical techniques to overcome these limitations.

The count measurements (*i.e.*, those made using PrivCount) include noise generated according to a normal distribution with mean zero and known variance. Thus for those measurements we compute confidence intervals (CIs) that include the true value with 95% probability. Moreover, for many values we wish to infer a network-wide total from our sample. We do so by dividing the reported values (and their CIs) by the fraction of the observations that our measuring relays make. For example, we measure 32 million streams using relays that comprise 1.5% of the exit weight and including added noise with a standard deviation of 3.1 million. We infer $(3.2e7 \pm 6.2e6)/0.015 = 2.1e9 \pm 4.1e8$ streams in the entire network, where the range provides a 95% confidence interval.

The unique-count measurements (*i.e.*, those made using PSC) include noise generated according to a binomial distribution with known parameters. In addition, hash-table collisions (which can occur within PSC's internal data storage [23]) can cause the measured value to be smaller than the true value. We adjust for these errors by computing 95% confidence intervals using an exact algorithm based on dynamic programming.

Extrapolating unique counts from our sample to the entire network can be a challenge. Unlike with standard counts, we need to know if the same items in our sample are seen elsewhere or not. For example, when counting unique domains it could be that the domains we count are each visited very frequently and thus are observed by all relays, which would indicate that our sample count is the same as the network-wide count. In several cases, we can handle this using other information we have about the frequency distribution of the observed items. For example, there is evidence that domains are visited following a power-law distribution [12, 32]. We also make some additional measurements to help determine likely parameters for these distributions. We can then construct confidence intervals for the network-wide unique counts by considering the probability of our local counts given different possible values for the overall count. We use Monte-Carlo simulations to determine these probabilities for more complicated distributions. Note that in some cases we cannot identify a likely frequency distribution. In these cases, with an observed value of x and a fraction p of all observations, we simply present the range of likely network-wide values to be $[x, x/p]$, where the lower end of the range covers the possibility that each item is frequently and the upper end covers the possibility of infrequent observations per item.

4 EXIT MEASUREMENTS

In this section, we present results and analyses of exit measurements taken from relays in a position to observe Tor's egress traffic.

4.1 Overview

Recall from §2.1 that Tor clients build *circuits* through a sequence of Tor relays over which they multiplex multiple *streams*. Each stream is associated with a TCP connection between an exit relay and a user-specified destination and is used to transfer data between that TCP connection and the client. For each stream, the client specifies a destination hostname or IP address which the exit relay requires in order to create a TCP connection on the client's behalf.

To better understand which web domains Tor users frequently visit, we instrument PrivCount and PSC to count the *initial* streams

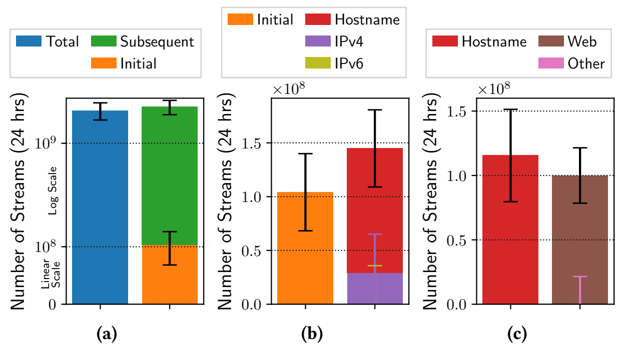

Figure 1: The number of streams of various types over 24 hours in Tor inferred from our exit relay observations, including 95% confidence intervals. In each sub-figure, the left bar shows the category total and the right stacked bar shows the breakdown of that category into more specific subcategories. Note that due to noise we measured a negative number of IPv6 (b) and Other (c) streams (which is why the corresponding bars are not visible in the chart); the most likely actual value of these counts is therefore 0.

that are created on circuits (*i.e.*, the first stream for each circuit) given that a *hostname* was provided in the stream by the client and the destination port requested by the client is a *web* port (either 80 or 443). We focus on initial streams because the Tor Browser uses a new circuit for each unique domain shown in the browser address bar. The initial stream on a circuit will therefore most directly indicate the user's intended destination, whereas subsequent streams that are created when loading a page to fetch embedded resources (*e.g.*, images, scripts, *etc.*) provide less useful indications of user intent. Second, we focus on streams that provide hostnames rather than IPv4 or IPv6 addresses because hostnames can be mapped to web domains much more easily than IP addresses. Finally, we are interested in web domains and so we do not measure domains on streams that request connection to destination ports not traditionally associated with web content.

4.2 Exit Stream Analysis

We first measure and analyze the breakdown of streams that provide a hostname and web port, compared to other types of streams. These measurements provide useful Tor usage information and also provide context about the overall fraction of Tor exit traffic that our domain measurements will cover.

Our PrivCount stream measurements were conducted between 2018-01-04 and 2018-01-05 during which the mean combined exit weight of our measurement relays (taken over the consensuses that were active during the measurement) was 1.5% of the total available exit weight in Tor. We infer the number of streams of various types over a 24 hour period in the entire Tor network using the observations from our relays and our fractional weight. The inferred values are shown in Figure 1 with 95% confidence intervals. Figure 1a shows that there are 2.1 billion (CI: [1.7; 2.5] billion) exit streams created in Tor in 24 hours, and only about 5% of those streams (104 million, CI: [68; 140] million) represent a circuit's first stream. Figure 1b shows that an insignificant number of these initial streams include an IPv4 or IPv6 address: our measured values for these stream subcategories were either negative or included 0 in the confidence interval (due to the added noise) and therefore it is possible that the actual value of each of the counters is zero. Figure 1c similarly shows that an insignificant number of initial streams containing a hostname target a non-web port (*i.e.*, a port other than 80 or 443). Our results confirm that almost all initial streams provide hostnames and target a web port (100 million, CI: [78; 121] million); the domain measurements that we describe next will focus on the hostnames provided in these streams.

4.3 Exit Domains

We now describe the results from our measurements of the number of domains observed in initial streams that also provide a hostname and a web port. To ease presentation, we refer to the measured values as *primary domains* or *domains* in the remainder of this section. Across our measurements, we construct sets of domain names and increment a counter for a set whenever we observe a primary domain that matches a domain name in that set.

Alexa Top Sites. We use the Alexa top 1 millions sites list[2] to help us understand which sites are visited by Tor users. We conducted two related PrivCount measurements. In the *Alexa rank* measurement, we sorted the sites by rank and split them into six sets of increasing size: set $i = 0$ contains the first 10^1 sites and set $i > 0$ contains the first 10^{i+1} sites excluding those in set $i - 1$. We used a separate set for torproject.org since early measurements revealed a significant number of accesses to that domain. The *Alexa rank* measurement was conducted between 2018-01-31 and 2018-02-01, during which our combined mean exit weight was 2.2%.

In the *Alexa siblings* measurement, we created a set for each of the top 10 sites in the Alexa list. For each such site we stripped the top level domain to produce a site basename (*e.g.*, google), and then added all entries from the top 1 million sites list that contained the basename into the corresponding set. We also used distinct sets for duckduckgo (rank 342, the default search engine in Tor Browser) and torproject (rank 10,244, developer of Tor Browser). As a result of this process, the google set (rank 1) was the largest (212 sites, including the rank 7 site google.co.in), while the reddit (rank 8) and qq (rank 9) sets were the smallest among top 10 sites containing 3 sites each (duckduckgo and torproject contained 1 site each). The *Alexa siblings* measurement was conducted between 2018-02-01 and 2018-02-02, during which our combined mean exit weight was 2.1%.

The results from the *Alexa rank* and *Alexa siblings* measurements are shown in Figure 2. We highlight three observations from these results. First, we observed torproject.org in 40.1% (CI: [39.9; 40.3]%) and 39.0% (CI: [38.8; 39.2]%) of primary domains. Surprised by this result, we conducted additional measurements and observed the domain onionoo.torproject.org in 43.4% (CI: [43.1; 43.7]%) of primary domains. Onionoo is a web service that provides other applications with access to Tor network status information. We contacted the Onionoo maintainers from The Tor Project, but we were unable to identify a clear reason for the high number of accesses to Onionoo from Tor.

Second, we observed sites in the amazon siblings set for 9.7% (CI: [9.5; 9.9]%) of primary domains, and sites in the google siblings set for 2.4% (CI: [2.2; 2.6]%) primary domains. To further explain

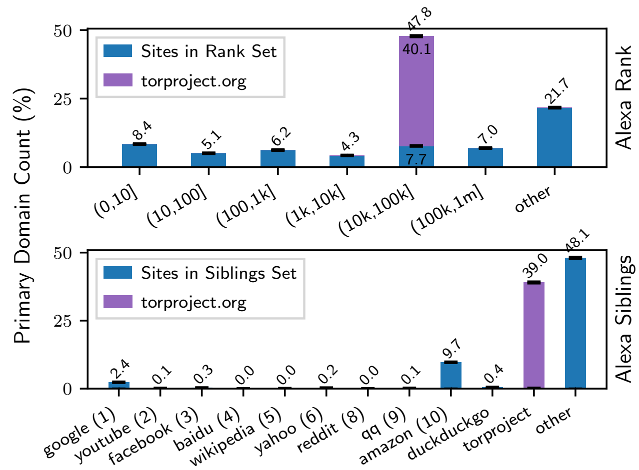

Figure 2: Results of PrivCount measurements of the frequency of membership of primary domains in subsets of the Alexa top 1 million sites list. The top plot shows subsets containing domains that are sorted by Alexa *rank*, and the bottom plot shows subsets containing domains of each of the top 10 sites as well as their *siblings*.

the unexpectedly high amazon result, we conducted an additional measurement during which we observed www.amazon.com in 8.6% (CI: [8.3; 8.9]%) of primary domains. We contacted Amazon employees but again were unable to find the reason for the relatively high number of accesses from Tor (here, due to lack of response).

Third, we observed that domains in the Alexa top sites list accounted for about 80% of all primary domains accessed from Tor, and that the smaller but more highly ranked domain sets have roughly equal frequency as the increasingly larger but more lowly ranked sets. We conclude from these results that the Alexa top sites list provides a reasonable representation of destinations visited by Tor users. The Alexa top sites list is often used as a destination model in Tor research, and most significantly in Tor website fingerprinting research [27]. Our results provide confirmation that using the top sites list as part of Tor research is appropriate.

Alexa Categories. We conducted a PrivCount measurement of primary domains by category (*e.g.*, news, science, sports, *etc.*) using Alexa category lists [3] which are limited to 50 sites per category. The measurement was conducted between 2018-01-29 and 2018-01-30, during which our mean combined exit weight was 2.1%. Additional insights from the measurement beyond those already presented are limited: the category containing amazon.com accounted for 7.6% (CI: [7.4; 7.8]%) of primary domains while 90.6% (CI: [90.3; 90.9]%) category (torproject.org was not in any of the categories).

Top-Level Domains. We measured the frequency with which all top-level domain (TLD) names that are contained in more than 10^4 entries in the Alexa top 1 million sites list also appeared in the observed primary domains. The measured TLDs include three main TLDs (.com, .org, and .net) as well as 11 country-specific TLDs. We conducted two related PrivCount TLD measurements, one in which

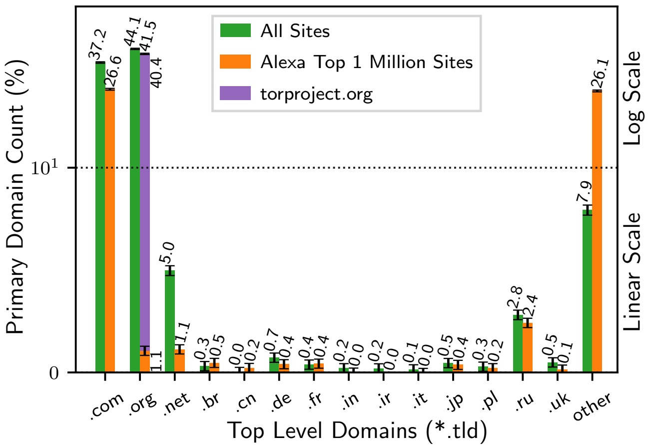

Figure 3: Results of PrivCount measurements of the frequency of membership of primary domains in top-level domain subsets constructed from *all sites* and from those sites present in the *Alexa top 1 million sites* list.

Table 2: Locally observed *unique* second level domain statistics measured using PSC.

Statistic	Count	95% CI
SLDs	471,228	[470,357; 472,099]
Alexa SLDs	35,660	[34,789; 37,393]

we count the TLDs of all primary domains using wildcards (conducted between 2018-02-02 and 2018-02-03 with 2.4% exit weight), and one in which we count the TLDs of only those primary domains that appear in the Alexa list (conducted between 2018-01-30 and 2018-01-31 with 2.3% exit weight). We measured torproject.org using a separate counter during the Alexa measurement, but our implementation of wildcard matching restricted us from doing so when measuring all sites.

The TLD measurement results are shown in Figure 3. Unsurprisingly, the three main TLDs make up the majority of the primary domains accessed by Tor users, though we note that .org would account for a significantly smaller fraction of domains without torproject.org. We found that the Russian TLD .ru accounted for 2.8% (CI: [2.6; 3.0]%) and 2.4% (CI: [2.2; 2.6]%) of country-specific TLDs, the largest observed by our relays. We also found that ~11% of .com, ~3% of .org, and ~4% of .net sites accessed in primary domains are not in the Alexa top sites list. Only 7.9% (CI: [7.7; 8.1]%) of TLDs from all primary domains did not match one of the TLDs that we measured, which increased to 26.1% (CI: [25.9; 26.3]%) of TLDs for primary domains in the Alexa top 1 million sites list.

Second-Level Domains. We conducted two measurements of the number of unique second-level domain (SLD) names that we observed in Tor primary domains. During both these measurements, we use only 5 out of our 6 exit relays (1.24% mean exit weight) in order to reduce operator overhead. We summarize our results in Table 2. In the first measurement, we measured the number of unique SLD names among those Tor primary domains which contain a TLD in the public suffix list [4]. From a measurement conducted between 2018-03-31 and 2018-04-01, we find that 471,228 (CI: [470,357; 472,099]) unique SLDs were accessed through our exits. We also measured the number of unique Alexa top 1 million SLDs (i.e. the SLDs of Alexa top 1 million sites) among those Tor primary domains. From a measurement conducted between 2018-03-24 and 2018-03-25, we infer that 35,660 (CI: [34,789; 37,393]) of such unique SLDs were accessed through our exits. From these results, we find that the unique count of accessed SLDs is more than ten times that of the unique count of Alexa top one million sites. From our measurements, we conclude that a long tail exists in the distribution of sites accessed over Tor.

To extrapolate the unique second-level domain measurements *for the entire Tor network*, we need to know the distribution of SLDs as seen by the exits. From previous research studies, we know that domains are visited following a power-law distribution [12, 32]. But we need additional measurements to determine the exponent of this power-law distribution. Therefore, we perform a series of simulations of clients visiting random destination sites (based on power-law distribution with random exponents) and construct confidence intervals for the network-wide unique SLD counts. We use the locally observed unique SLDs count as a self-check.

This method appears to work well for the unique Alexa top 1 million SLDs. The results of 100 simulations reveals an inferred network-wide unique count of 513,342 (CI: [512,760; 514,693]) accesses to the Alexa top 1 million list. That is, slightly more than half of the Alexa top sites are accessed over Tor over 24 hours. Unfortunately, the inability to closely fit SLD accesses to a distribution prevented us from using this approach to extrapolate the number of network-wide SLD accesses.

5 CLIENT MEASUREMENTS

We conducted a number of measurements to better understand *who* uses Tor and to determine *how many* users access the network.

5.1 Tor Connections and Clients

We first measure the number of client connections using our PrivCount deployment. Over the 24 hour measurement period beginning on 2018-04-07, the mean probability of selecting our relays in the entry position was 0.0144. Since clients typically choose their guards according to guard weights, we can project our local PrivCount measurements to infer Tor-wide results by dividing our observed counts by 0.0144. We summarize our results in Table 4.

The amount of daily data being transferred across Tor is 517 TiB (CI: [504; 530] TiB), which represents the sum of client uploads and downloads. We note that this includes Tor cell overheads, so the actual amount of client payload data transferred would be 2-3% less.

We find that there are about 148 million (CI: [143; 153] million) client connections, through which 1,286 million client circuits (CI: [1,246; 1,326] million) are multiplexed, per day, across the Tor network. This is significantly higher than the 80.6 million client connections reported by Jansen and Johnson [26] two years ago. We suspect that the discrepancy is due to ongoing distributed denial-of-service (DDoS) attacks that began affecting the Tor network back in December 2017 [24].

Tor Clients. We next consider the number of *unique* clients access the Tor network. Here, we use PSC to safely capture the approximate number of clients observed by our guard relays. Unlike existing work that also attempts to quantify the number of Tor users [14, 26, 34, 38], we do not store (even temporarily) IP addresses since PSC uses oblivious counters. Similar to previous studies, we assume that there is a one-to-one mapping between client IPs and unique Tor clients, although this may be violated, for example, by mobile users with changing IP addresses or by clients behind Network Address Translation (NAT). In addition, we count bridges as clients, as their identities are private.

We use PSC to collect data from our relays that have a non-zero guard probability, for a 24 hour period beginning on 2018-04-14 during which our guards had a combined weight of 1.19%. Measurement results are reported in Table 5. We observe over 313 thousand unique client IPs using our guards (this experiment used two CPs due to the temporary unavailability of one). This is a surprisingly high number, given that Tor Metrics [43] (using a very different and unvalidated estimation technique) reports 2.15 million clients per day in April 2018 *for the entire Tor network*. We would expect a typical client to connect to three guards (clients currently use one guard for data but two additional guards for directory updates [7]), and so we would expect to see closer to $0.0119 \times 3 \times 2.15e6 = 76,755$ unique clients IPs. To the extent that each unique IP observed in a 24-hour period represents a new user, this suggests that Tor is underestimating its total number of users by a factor of 4, and the total number of daily users is closer to $313,213/0.0119/3 \approx 8,773,473$.

To better understand the network-wide number of unique client IPs in Tor, we perform additional PSC measurements using sets of relays of *different sizes*. We can then compare how the count grows with the measuring set to its predicted growth under different client models to identify the most likely model. In particular we would like to validate a model of how many guards each client contacts, which we expect to typically be 3, but could be less or more for several reasons, including guard/exit conflicts, multiple clients behind a single NAT IP, Tor bridges serving many clients, and tor2web instances [11]. We made two 24-hour measurements, one starting on 2018-05-12 using DCs with 0.42% of the guard weight and the other starting on 2018-05-13 using a disjoint set of DCs with 0.88% of the guard weight. We counted 148,174 (CI: [148; 161] thousand) and 269,795 (CI: [269; 315] thousand) unique client IPs during these measurements, respectively. Observe that the latter number is significantly smaller than we would expect if each client contacted only one guard, in which case we would predict $148,174 \times (0.88/0.42) \approx 310,460$. This indicates that indeed client IPs typically connect to multiple guards.

To identify the number of guards that typical client IPs connect to, we consider a range of guards per client and analyze the resulting consistency with our measurements. Let g denote the number of guards each client connects to in the model. We use simulation to extrapolate from our two measurements two separate confidence intervals for the network-wide unique client IP counts. We discover that these CIs are *disjoint* unless g is in the range $[27, 34]$. This is a much higher value of g than should be the case for typical clients, and we believe that this indicates that this is a poor model of how clients connect to guards. It does imply, however, that there are some clients IPs that are connected to many more guards than is typical.

Table 3: Network-wide promiscuous clients and client IPs

Guards per client	Promiscuous clients 95% CI	Network-wide client IPs 95% CI
3	[15,856; 21,522]	[10,851,783; 11,240,709]
4	[15,129; 21,056]	[8,195,072; 8,493,863]
5	[14,428; 20,451]	[6,605,713; 6,849,612]

Thus we consider a refinement on this model in which a set of "promiscuous" clients connects to *all* guards in a 24-hour period, and the remaining "selective" clients connect to g guards only. The promiscuous clients capture the likely behavior of Tor bridges, tor2web clients, and clients behind a NAT IP. We then determine a range for the number p of promiscuous clients by considering values of g that are likely given our understanding of the Tor protocol: $g \in \{3, 4, 5\}$ (all clients should connect to 3 guards for directory updates, some clients may connect to 1 or 2 more due to guard churn). For each of these values of g, there were some values for the number p of promiscuous clients that was consistent with our two client-IP measurements, that is, that yielded a non-empty intersection of the two CIs. Table 3 shows these ranges, which indicates about 14–22 thousand promiscuous clients. Observe that bridges alone seem unlikely to make up this population, as this range is much larger than the approximately $1,640$ bridges reported by Tor Metrics at the time of our measurements, and in April 2016 Matic et al. [37] only discovered 50% more bridges than reported by Tor Metrics. Table 3 also shows confidence intervals for the number of network-wide client IPs, calculated for each g as the union of CIs over all p in the range shown. For what we believe is the most likely number of guards per selective client ($g = 3$), our measurements indicate about 11 million unique client IPs network-wide, which suggests a true value over 5 times the number of users estimated by Tor Metrics. Even for $g = 5$, the range suggests over 3 times as many daily users as Tor currently estimates.

Client churn. Tor clients may go offline, and we expect that the Tor network experiences a high degree of client *churn* (*i.e.*, change in the connected client population). Using PSC, we recorded the count of unique client IPs over consecutive one-day, four-day, and seven-day period measurements beginning on 2018-07-02. We observed 306,905 (CI: [306,731; 343,832]), 670,955 (CI: [670,526; 890,786]), and 885,778 (CI: [885,088; 1,341,759]) unique client IPs, respectively. Comparing these results (see Table 5), we can conclude that the client churn rate is not constant. The client churn rate is on average 121,350 (CI: [121,265; 182,318]) client IPs per day for the first three days, and 71,608 (CI: [71,521; 150,324]) client IPs per day for the next three days.

We observe that the client churn rate decreases. We can consider a few possible explanations for this. The first is that some decreasing number of "new" clients connects to our guards everyday. This seems unlikely, as even if the rate of new users joining Tor has some periodic behavior, we observed a decreasing churn rate with two consecutive measurements of different lengths. A more likely explanation is that some clients (*e.g.*, mobile users) have a small

Table 4: Network-wide client usage statistics, inferred from PrivCount measurements.

Statistic	Count	95% CI
Data (TiB)	517	± [504; 530]
Connections ($\times 10^6$)	148	± [143; 153]
Circuits ($\times 10^6$)	1,286	± [1,246; 1,326]

Table 5: Locally observed *unique* client statistics measured using PSC.

Statistic	Count	95% CI
IPs	313,213	[313,039; 376,343]
Countries	203	[141; 250]
ASes	11,882	[11,708; 12,053]
IPs (1-day)	306,905	[306,731; 343,832]
IPs (4-day)	670,955	[670,526; 890,786]
Churn (days 1-4)	121,350/day	[121,265; 182,318]/day
IPs (7-day)	885,778	[885,088; 1,341,759]
Churn (days 4-7)	71,608/day	[71,521; 150,324]/day

pool of IP addresses from which they connect to Tor. Over time, their guards observe an increasing fraction of that pool, and the chance that an unseen IP address is used decreases. Another possible explanation is that there is some number of promiscuous clients, but they take longer than a day to connect to each guard at least once.

5.2 Client Composition and Diversity

We additionally explore the geopolitical distribution of Tor clients. We resolve each client IP address to its host country using the MaxMind GeoLite2 country database, and use PrivCount to count the number of client connections, bytes transferred, and circuits created, broken down by country. For most of the world's 250 countries, we were unable to determine useful counts since the added noise (to achieve differential privacy) overwhelm the actual count. We show the countries with significant inferred counts in Figure 4.

The United States (US), Russia (RU), and Germany (DE) have both the greatest client connection counts and data transfer amounts (see Figure 4, *left* and *center*). Assuming that the distribution of client connection counts reflects the distribution of client usage, we can surmise that these three countries use Tor the most.

Interestingly, our results disagree with those from the Tor Metrics Portal, which ranks the United Arab Emirates (UAE) as contributing the second largest number of Tor users; in contrast, we did not find the UAE to be among the most significant contributors. Although we cannot fully explain this discrepancy, we are somewhat skeptical of the Tor Metric Portal's results: the value reported by the Tor Metrics Portal implies that nearly 4% of the entire population of the UAE accesses Tor on a daily basis, which seems unlikely. However, although the UAE has neither a significant client connection count nor client data transferred, it ranks sixth in the number of client circuits (see Figure 4, *right*; marked as "AE"). One intriguing possibility is that the majority of the Tor clients from the UAE are partially blocked from using Tor: while they are able to construct directory circuits (recall that Tor's user statistics are estimated from the number of directory requests), they are prevented from establishing regular Tor circuits, causing the Tor software to repeatedly perform directory fetches. We leave the exploration of this strange Tor client behavior in the UAE as a future research direction.

To determine whether Tor has a global userbase, we use PSC to count the number of unique countries from which clients originate. Since the actual count can only be at most 250 (*i.e.*, the total number of countries worldwide), invariably the differentially private noise overwhelms the actual count. Therefore to reduce the effects of noise, we average the country counts between two consecutive one-day measurements, beginning on 2018-05-09. We find that clients from 203 (CI: [141; 250]) different countries access Tor. Comparing our results to earlier studies from McCoy et al. [38] (2008) and Chaabane et al. [14] (from 2010) that report clients from approximately 125 countries, we can conclude that the locations from which Tor is accessed has significantly diversified in the past decade.

We were unable to extrapolate the unique country count for the entire Tor network since the frequency distribution of the countries, as seen by the guards, is difficult to determine. However, we can still roughly estimate the range of network-wide values to be [141, 250], where the lower bound is the least possible observed count and the upper bound is the total number of countries that exists worldwide.

Network Diversity. We also measure the network diversity of Tor users by mapping each client IP address to its autonomous system (AS) using the IPv4 and IPv6 datasets (dated 26th November 2017) from CAIDA [6], consisting of 59,597 ASes. Using PSC, we first measure the unique AS count of the client IPs, for a 24 hour period beginning on 2018-04-18. We observe clients from about 11,882 ASes (CI: [11,708; 12,053])—approximately 20% of the number of defined ASes.

To extrapolate the unique AS count *for the entire Tor network*, we must first identify the frequency distribution of the ASes, as seen by the guards. This is challenging. Therefore, we present the range of most-likely network-wide values as [11,708; 59,597], where the lower bound is the least possible observed count and the upper bound is the total number of ASes.

To determine whether there are any "hotspot" ASes, we run PrivCount and record the client connection count for each AS. Since the total number of ASes (59,597) is large, the per-AS differentially-private noise is by chance large for some of them. Therefore to reduce the effect of noise, we run the measurement in two phases, each consisting of four consecutive one-day measurements, and sum the measurements over days. In the first phase, beginning on 2018-06-22, we include all ASes, and in the second phase, beginning on 2018-06-27, we include only those 1,653 ASes that did not have zero in their confidence interval in at least one of the days in the first phase. Thus the first phase acts as a filter to reduce the ASes under consideration to those with likely positive true counts. We average the counts for the 1,653 ASes across all eight days, and plot the network-wide inferences (with 1.2% mean entry weight) of the resulting top 10 ASes in Figure 5. Almost 22% of Tor's entry connections are confined to these 10 ASes. A surprising number of

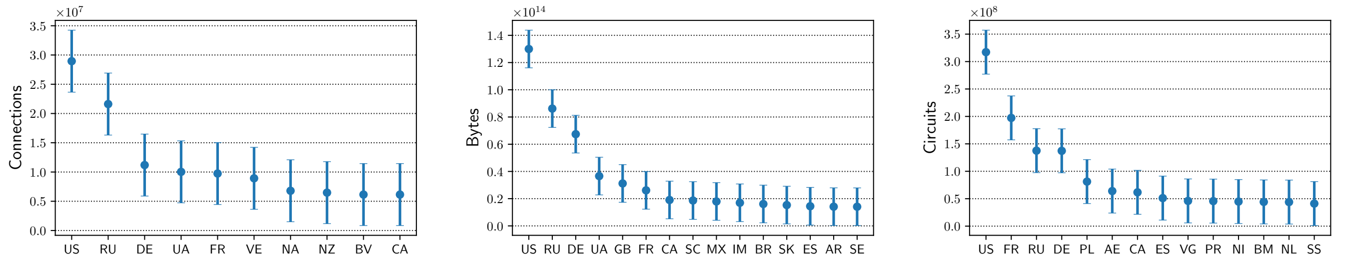

Figure 4: Tor per country client usage statistics inferred from PrivCount measurements - the counts of client connections (*left*), amount of client data transferred (*center*), and the counts of client circuits (*right*). Error bars show 95% confidence intervals.

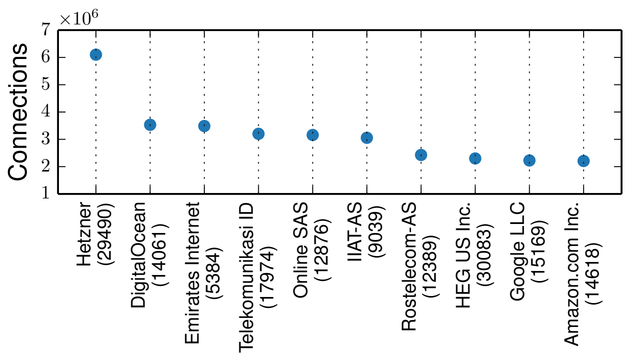

Figure 5: Top 10 ASes' client connection count inferred from PrivCount. Error bars with 95% confidence intervals are very close to the plotted value, and are not visually discernable.

Table 6: Network-wide number of *unique* onion addresses inferred from our HSDir PSC measurements.

Statistic	Count	95% CI
Addresses published	70,826	[65,738; 76,350]
Addresses fetched	74,900	[34,363; 696,255]

these ASes are hosting providers (*e.g.*, Hetzner, DigitalOcean, Online SAS), from which we wouldn't expect consumer connections. We suspect that these entry connections are from a mix of Tor bridges, onion services, Tor2web proxies, and Tor network scanners.

6 ONION-SERVICE MEASUREMENTS

We conducted a number of measurements to determine *how many* Tor onion services exist, *which kinds* of services are popular, and *how* these services are used.

6.1 Unique Onion Addresses

Recall from §2.1 that Tor onion services store their descriptors at onion service directories (HSDirs) in a DHT. To measure the number of *unique* onion services in the Tor network, we instrument our HSDirs to report the onion service *address* for each version 2 (v2) [8] descriptor that is *published* to and *fetched* from the DHT. (We don't measure version 3 (v3) onion service descriptors because the onion address is obscured using key blinding [9].) We use the PSC deployment described in §3.1 to safely capture the approximate number of v2 onion services observed by our HSDirs. Unlike existing work that also attempts to quantify the number of onion services [25, 29, 30, 39], we avoid the need to store (even temporarily) onion addresses, since PSC uses oblivious counters. Table 6 summarizes our onion service address measurements.

Unique Onion Service Addresses Published. From a PSC measurement conducted between 2018-04-23 and 2018-04-24 during which our mean combined HSDir publish weight was 2.75%, we observed 3,900 (CI: [3,769; 4,045]) unique v2 onion addresses in descriptors published to our HSDirs. We extrapolate these results based on HSDir replication to infer that the total number of unique onion service addresses published *for the entire Tor network* is 70,826 (CI: [65,738; 76,350]). Accordingly, our relays observed at least 4.93% of Tor's unique onion addresses. Using a different extrapolation method [25], the Tor Metrics Portal [43] estimates a total of 79 thousand unique v2 onion services published in the Tor network at the time of our measurement. The Tor metrics estimate does not include a confidence interval which we expect to be significant since *every* reporting relay adds noise (due to the lack of secure aggregation of Tor metrics reports). Therefore, we expect that our confidence intervals would overlap.

Unique Onion Service Addresses Fetched. From a PSC measurement conducted between 2018-04-29 and 2018-04-30 during which our mean combined HSDir fetch weight was 0.534%, we observed 2,401 (CI: [1,101; 3,718]) unique v2 onion addresses in descriptors fetched from our HSDirs. We extrapolate these results based on HSDir replication to infer that the total number of unique onion service addresses fetched *for the entire Tor network* is 74,900 (CI: [34,363; 696,255]). By comparing the number of unique onion addresses published and fetched, we estimate that between 45% and 100% of active onion services are used by clients. Note that the Tor Metrics Portal [43] does not estimate the number of unique onion addresses fetched by clients, and its estimate of the number of unique v2 onion addresses published occasionally varies significantly (*e.g.*, it increased by 40 thousand during our measurement).

6.2 Onion Service Directory Usage

While the number of unique onion *addresses* (§6.1) informs our understanding of how many onion services exist, the number of *descriptor* actions informs our understanding of usage. Therefore, we

Table 7: Network-wide onion service descriptor statistics inferred from our HSDir PrivCount measurements.

Statistic	Count	95% CI
Fetched	134 million	[117; 150] million
Succeeded	12.2 million	[10.6; 13.7] million
Failed	121 million	[103; 140] million
Fail rate	1,400 failed/s	[1,192; 1,620] failed/s
Public	56.8%	[36.9; 83.6]%
Unknown	47.6%	[28.8; 72.7]%

use the PrivCount deployment described in §3.1 to safely measure the number of v2 onion service descriptors that are *fetched* from our HSDirs. We also count the number of requests for descriptors that are not stored in the DHT, typically because the service is inactive. Table 7 summarizes our onion service descriptor measurements.

Onion Service Descriptors Fetched. From a PrivCount measurement conducted between 2018-05-20 and 2018-05-21 during which our mean combined HSDir fetch weight was 0.465%, we infer that there were a total of 134 million (CI: [117; 150] million) v2 descriptors fetched *in the entire Tor network*. Surprisingly, 90.9% (CI: [87.8; 93.2]%) of these fetches failed, *i.e.*, the corresponding descriptor was not present in the HSDir's cache or the request was malformed. Our results indicate that there are *at least* 103 million failures per day, or about 1,192 failures per second. To be sure we did not observe a network anomaly, we conducted this measurement multiple times and observed consistent results. We speculate that this large failure rate may be due to botnets or onion site scanners with outdated onion address lists. Tor's current statistics reporting infrastructure is unable to collect HSDir lookup failures [28], but we have demonstrated the feasibility of using the PrivCount protocol to collect this statistic in a privacy-preserving manner.

From the measurement described above, we infer that there were a total of 12.2 million (CI: [10.6; 13.7] million) v2 descriptors successfully fetched *in the entire Tor network*. For every successful descriptor fetch, we checked if the address was available in the latest ahmia onion site search index [1]. We found that 56.8% (CI: [36.9; 83.6]%) of descriptors fetched were present in the ahmia list at the time of our measurement, while 47.6% (CI: [28.8; 72.7]%) were not. Our results indicate that a majority of successful onion service accesses are visits to onion sites with publicly available addresses.

6.3 Rendezvous Point Usage

Recall from §2.1 that in order for a client to communicate with an onion service, both parties build circuits to a relay called a rendezvous point (RP), and that this rendezvous circuit carries all end-to-end encrypted application payloads. While we cannot measure onion service usage by counting streams (onion service streams are unobservable by our RPs because data cells are end-to-end encrypted), we can infer rendezvous circuit activity by counting the number of cells on the circuit. We instrument our RPs to report the number of cells sent on v2 [8] and v3 [9] rendezvous circuits, and use the PrivCount deployment described in §3.1 to safely measure the approximate number of onion service rendezvous circuits and cells relayed at our RPs. Table 8 summarizes our RP measurements.

Table 8: Network-wide rendezvous statistics inferred from our RP PrivCount measurements.

Statistic	Count	95% CI
Total Circuits	366 million	[351; 380] million
Succeeded	8.08%	[3.47; 13.1]%
Failed: conn. closed	4.37%	[0.0; 9.23]%
Failed: circuit expired	84.9%	[77.0; 93.5]%
Cell payload	20.1 TiB	[15.2; 24.9] TiB
Cell payload / second	2.04 Gbit/s	[1.55; 2.53] Gbit/s
Cell payload / circuit	730 KiB/circ.	[341; 2,070] KiB/circ.

Rendezvous Circuit Activity. From a PrivCount measurement conducted between 2018-05-22 and 2018-05-23 during which our mean combined rendezvous weight was 0.88%, we infer that there were a total of 366 million (CI: [351; 380] million) rendezvous circuits *in the entire Tor network*. Note that since a successful rendezvous involves a client and service circuit, each such rendezvous is counted as 2 circuits at the RP. Surprisingly, only 8.08% (CI: [3.47; 13.1]%) of the circuits that we observed succeeded and were active, *i.e.*, they were used to transfer at least one cell containing an application payload. We found that 4.37% (CI: [0.0; 9.23]%) of the observed rendezvous circuits failed because the connection to the RP was closed before the service completed the rendezvous protocol, and 84.9% (CI: [77.0; 93.5]%) of the observed rendezvous circuits failed because the circuit expired (timed-out) before the service completed the rendezvous protocol.

Rendezvous Circuit Data. From the same PrivCount measurement as above, we infer a total of 20.1 TiB (CI: [15.2; 24.9] TiB) of cell payload data on rendezvous circuits *in the entire Tor network*. Our inference corresponds to a mean of 2.04 Gbit/s (CI: [1.55; 2.53] Gbit/s) of payload data. By combining our circuit and cell observations, we calculate that the mean amount of data per active Tor rendezvous circuit is 730 KiB (CI:[341; 2070] KiB).

7 RELATED WORK

Several efforts have attempted to improve our understanding of the live Tor network. Most notably, the Tor Project has allowed researchers to access aggregate and longitudinal data about the Tor network through its Tor Metrics Portal [43] since 2010, with directory data spanning back to May 2004 [33].

Tor Metrics Portal. The Metrics Portal indirectly estimates the number of Tor users by counting the number of requests to the subset of Tor directory mirrors that participate in statistics collection, and then extrapolating to the entire network by dividing by the fraction of participating directory mirrors. This technique was originally proposed by Loesing *et al.* [34]. In contrast with the Tor Metrics Portal, our measurements are based on *direct* observations of connecting users and do not depend on heuristics about how often clients access directory mirrors. (Indeed, as described in §5, our direct measurements do not align well with the Metric Portal's indirect estimates.)

Towards safe Tor measurements. McCoy *et al.* performed one of the first studies that attempted to discern *how* users used Tor [38].

There, the authors performed packet capture at a small subset of Tor ingress and egress points to determine the distributions of user locations and exit traffic by port (*i.e.*, by application). Chaabane *et al.* used similar methods to examine BitTorrent behavior on Tor [14]. Approaches such as these that collect unobfuscated and potentially sensitive network traces have been criticized on both ethical and legal grounds [40, 41].

There has been an increasing effort in developing techniques to *safely* measure the Tor network. Elahi et al. [22] were the first to propose the use of differential privacy to provide privacy-preserving aggregate data about Tor. Specifically, they propose two schemes known collectively as PrivEx. PrivCount [26], which we use in our measurements, extends the secret-sharing variant of PrivEx by supporting repeatable measurement phases. HisTorϵ [35] also proposes the use of differential privacy to safely measure statistics on anonymity networks, and adds integrity protections by bounding the influence of malicious data collectors (DCs). However, compared to PrivCount, HisTorϵ is limited in the types of queries that it supports and lacks PrivCount's mature implementation. The private set-union cardinality (PSC) protocol of Fenske *et al.* [23] also provides differentially private measurements. We review differential privacy in §2.2. Sections §2.3 and §2.4 describe PrivCount and PSC in more detail.

Understanding Onion Services. Goulet *et al.* [25] describe the benefits and privacy risks of statistics collection for onion services. They conclude with suggestions for privacy-preserving statistics collection, including the use of differential privacy. Their proposal partly inspired our work, in which we use differential privacy and other techniques [23] to conduct measurements of onion sites. Biryukov *et al.* demonstrate how the design of onion services allows an attacker to gauge the popularity of an onion service and potentially deanonymize it [13]. However, their envisioned attacks are not suitable for conducting privacy-preserving measurements. Recently, Jansen *et al.* show how to use PrivCount to safely measure the Tor network to discover the popularity of onion services [27]. Their work focused on traffic analysis attacks and the popularity study of a single social networking onion service. Finally, Owen and Savage perform empirical measurements of Tor's onion services [39]. We apply similar techniques—operating Tor relays and observing HSDir lookups—but also protect user privacy by using differentially private techniques.

8 ETHICAL AND SAFETY CONSIDERATIONS

Statistics collection in the setting of an anonymity network inherently carries risk since the exposure of information could degrade the privacy of the network's users and potentially subject them to harm. Guiding our study were the four criteria for ethical network research established by the Menlo Report [18]: we uphold the principle of *respect for persons* through the careful and principled application of data protections, including the use of differentially private techniques and the avoidance of collecting personally identifiable information (e.g., IP addresses). Following the principle of *beneficence*, we balance the (low) risk of harm with the potential benefits of the research—namely, an increased understanding of the Tor network that could inform research on improving the network's security and performance. Our techniques achieve the Menlo Report's notion of *justice*, since our statistics do not target a specific subpopulation of Tor's users. Finally, we achieve *respect for law and public interest* through transparency in our methods: we use open source tools [23, 26] for statistics collection and we submitted our research plan (prior to its implementation) to several review bodies. We presented it to the Tor Research Safety Board [10], which concluded that our plan "provides a plausible strategy for safely measuring trends on the Tor network."[2]. (Although some of the authors serve on the TRSB, we were not involved in the TRSB's deliberations of our research plan.) Our measurement study was approved by the ethics board at the University of New South Wales and certified as being exempt as non-human subjects research by the Institutional Review Board (IRB) at Georgetown University Additionally, aspects of this study were discussed with members of Georgetown's Office of General Counsel, who raised no concerns.

9 CONCLUSION AND DISCUSSION

This paper presents the most comprehensive privacy-preserving measurement study of the live Tor network to date. Our findings confirm that Tor is used predominantly for web browsing and that Tor users tend to visit the same popular (*i.e.*, top Alexa) sites as do ordinary (and non-anonymous) Internet users. We observe a surprisingly large number of Tor connections and unique client IP addresses, suggesting that the heuristically-derived estimates from the Tor Metrics Portal are significantly underreporting Tor's userbase. We find that the network's clients are highly distributed, connecting from more than 200 countries and nearly 12,000 ASes. These clients construct more than 1.2 billion anonymous circuits per day, carrying approximately 517 TiB of data (6.1GiB/s). Unexpectedly, our measurements show that more than 90% of onion service descriptor fetches fail, suggesting the presence of botnets on Tor or aggressive onion site crawlers with outdated address lists. We report 20 TiB of data transferred daily over circuits to onion services, representing roughly 3.9% of all Tor traffic. Some of the data (for our findings) is available for download at https://security.cs.georgetown.edu/measurement-study/.

Our study does not attempt to distinguish between human-driven and automated activity on the Tor network. An intriguing open problem is the construction of techniques to identify when measurements are heavily influenced by automated activities (for example, when swarms of infected hosts belonging to a botnet connect to Tor). Indeed, the enormously high failure rate of onion service descriptor fetches strongly suggests that automated behavior likely does occur on Tor. Our hope is that measurement studies such as this will help inform developers and security researchers of unexpected automated activities — a necessary prerequisite to understanding and potentially defending against malicious automated processes. Moreover, our findings can also aid in developing better Tor network traffic and user behavior models that can be used to improve Tor's performance and security.

ACKNOWLEDGMENTS

We thank the anonymous TPC and Shadow PC reviewers for their valuable comments, and especially Hamed Haddadi for helping

[2]The full text of the TRSB's findings is available [36]

shepherd this paper. We also thank Matthew Traudt for running data collectors for our measurement. This work has been partially supported by the Office of Naval Research, the National Science Foundation under grant numbers CNS-1149832 and CNS-1527401, and the Department of Homeland Security Science and Technology Directorate, Homeland Security Advanced Research Projects Agency, Cyber Security Division under agreement number FTCY1500057. The views expressed in this work are strictly those of the authors and do not necessarily reflect the official policy or position of any employer or funding agency.

REFERENCES

[1] Ahmia. https://ahmia.fi.
[2] Alexa Top 1 Million Global Sites. https://www.alexa.com/topsites. Data available at http://s3.amazonaws.com/alexa-static/top-1m.csv.zip. Accessed 2017-12-21.
[3] Alexa Top Sites by Category. https://www.alexa.com/topsites/categories. Accessed 2018-01-11.
[4] Public Suffix List. https://publicsuffix.org.
[5] Ricochet. https://ricochet.im/.
[6] Routeviews Prefix-to-AS mappings (pfx2as) for IPv4 and IPv6. http://data.caida.org/datasets/routing/routeviews-prefix2as/.
[7] Tor directory protocol, version 3. https://gitweb.torproject.org/torspec.git/plain/dir-spec.txt.
[8] Tor Rendezvous Specification (version 2). https://gitweb.torproject.org/torspec.git/plain/rend-spec-v2.txt.
[9] Tor Rendezvous Specification (version 3). https://gitweb.torproject.org/torspec.git/plain/rend-spec-v3.txt.
[10] Tor Research Safety Board. https://research.torproject.org/safetyboard.html.
[11] Tor2web: Browse the tor onion services. https://www.tor2web.org/.
[12] Lada A Adamic and Bernardo A Huberman. Zipf's Law and the Internet. *Glottometrics*, 3(1):143–150, 2002.
[13] Alex Biryukov, Ivan Pustogarov, and Ralf-Philipp Weinmann. Trawling for Tor Hidden Services: Detection, Measurement, Deanonymization. In *IEEE Symposium on Security and Privacy (Oakland)*, 2013.
[14] A. Chaabane, P. Manils, and M.A. Kaafar. Digging into Anonymous Traffic: A Deep Analysis of the Tor Anonymizing Network. In *IEEE Network and System Security*, 2010.
[15] Thomas M. Dalton. Affidavit of Special Agent Thomas M. Dalton. Available at https://cbsboston.files.wordpress.com/2013/12/kimeldoharvard.pdf.
[16] Roger Dingledine and Nick Mathewson. Tor Protocol Specification. https://spec.torproject.org/tor-spec.
[17] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-Generation Onion Router. In *USENIX Security*, 2004.
[18] David Dittrich, Erin Kenneally, et al. The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research. Technical report, US Department of Homeland Security, August 2012.
[19] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our Data, Ourselves: Privacy via Distributed Noise Generation. In *Advances in Cryptology (EUROCRYPT)*, 2006.
[20] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating Noise to Sensitivity in Private Data Analysis. In *Theory of Cryptography Conference*, 2006.
[21] Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.
[22] Tariq Elahi, George Danezis, and Ian Goldberg. PrivEx: Private Collection of Traffic Statistics for Anonymous Communication Networks. In *ACM Conference on Computer and Communications Security (CCS)*, 2014.
[23] Ellis Fenske, Akshaya Mani, Aaron Johnson, and Micah Sherr. Distributed Measurement with Private Set-Union Cardinality. In *ACM Conference on Computer and Communications Security (CCS)*, 2017.
[24] David Goulet. tor-project listserv post, available at https://lists.torproject.org/pipermail/tor-project/2017-December/001604.html, 2017.
[25] David Goulet, Aaron Johnson, George Kadianakis, and Karsten Loesing. Hidden-service Statistics Reported by Relays. Technical Report 2015-04-001, The Tor Project, Inc., April 2015.
[26] Rob Jansen and Aaron Johnson. Safely Measuring Tor. In *ACM Conference on Computer and Communications Security (CCS)*, 2016.
[27] Rob Jansen, Marc Juarez, Rafael Galvez, Tariq Elahi, and Claudia Diaz. Inside Job: Applying Traffic Analysis to Measure Tor from Within. In *Network and Distributed System Security Symposium (NDSS)*, 2018.
[28] George Kadianakis. Some Statistics about Onions. https://blog.torproject.org/some-statistics-about-onions.
[29] George Kadianakis, David Goulet, Karsten Loesing, and Aaron Johnson. Better Hidden Service Stats from Tor Relays. https://gitweb.torproject.org/torspec.git/tree/proposals/238-hs-relay-stats.txt, 2014.
[30] George Kadianakis and Karsten Loesing. Extrapolating Network Totals from Hidden-Service Statistics. Technical Report 2015-01-001, The Tor Project, January 2015.
[31] David Karger, Eric Lehman, Tom Leighton, Rina Panigrahy, Matthew Levine, and Daniel Lewin. Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web. In *ACM Symposium on Theory of Computing (STOC)*, 1997.
[32] Serge A Krashakov, Anton B Teslyuk, and Lev N Shchur. On the Universality of Rank Distributions of Website Popularity. *Computer Networks*, 50(11):1769–1780, 2006.
[33] Karsten Loesing. 10 Years of Collecting Tor Directory Data. Tor Blog post. Available at https://blog.torproject.org/10-years-collecting-tor-directory-data.
[34] Karsten Loesing, Steven J. Murdoch, and Roger Dingledine. A Case Study on Measuring Statistical Data in the Tor Anonymity Network. In *Financial Cryptograpy and Data Security*, 2010.
[35] Akshaya Mani and Micah Sherr. Historϵ: Differentially Private and Robust Statistics Collection for Tor. In *Network and Distributed System Security Symposium (NDSS)*, 2017.
[36] Akshaya Mani, T Wilson-Brown, Rob Jansen, Aaron Johnson, and Micah Sherr. Understanding Tor Usage with Privacy-Preserving Measurement: Auxiliary Material. https://dl.acm.org/citation.cfm?id=3278549, 2018.
[37] Srdjan Matic, Carmela Troncoso, and Juan Caballero. Dissecting Tor Bridges: a Security Evaluation of Their Private and Public Infrastructures. In *Network and Distributed Systems Security Symposium*, pages 1–15, 2017.
[38] Damon McCoy, Kevin Bauer, Dirk Grunwald, Tadayoshi Kohno, and Douglas Sicker. Shining Light in Dark Places: Understanding the Tor Network. In *Privacy Enhancing Technologies Symposium (PETS)*, 2008.
[39] Gareth Owen and Nick Savage. Empirical Analysis of Tor Hidden Services. *IET Information Security*, 10(3):113–118, 2016.
[40] Chris Soghoian. Researchers Could Face Legal Risks for Network Snooping. http://www.cnet.com/news/researchers-could-face-legal-risks-for-network-snooping/, July 2008.
[41] Christopher Soghoian. Enforced Community Standards for Research on Users of the Tor Anonymity Network. In *Workshop on Ethics in Computer Security Research (WECSR)*, 2011.
[42] Tor: Onion Service Protocol. https://www.torproject.org/docs/onion-services.html.en.
[43] Tor Project, Inc. Tor Metrics Portal. https://metrics.torproject.org/.