Dissecting Tor Bridges: a Security Evaluation of Their Private and Public Infrastructures

Srdjan Matic*†, Carmela Troncoso†, Juan Caballero†
†IMDEA Software Institute *Universita degli Studi di Milano
{srdjan.matic, carmela.troncoso, juan.caballero}@imdea.org

***Abstract*—Bridges are onion routers in the Tor Network whose IP addresses are not public. So far, no global security analysis of Tor bridges has been performed. Leveraging public data sources, and two known Tor issues, we perform the first systematic study on the security of the Tor bridges infrastructure. Our study covers both the public infrastructure available to all Tor users, and the previously unreported private infrastructure, comprising private nodes for the exclusive use of those who know their existence.**

Our analysis of the public infrastructure is twofold. First, we examine the security implications of the public data in the CollecTor service, identifying several pieces of data that may be detrimental for the security of bridges. Then, we measure security relevant properties of public bridges. Our results show that the 55% of public bridges that carry clients are vulnerable to aggressive blocking; that 90% of bridge clients use default bridges that are trivial to identify; that the concurrent deployment of Pluggable Transports in bridges reduces the security of the most secure transports; and that running non-Tor services in the same host as a bridge may harm its anonymity.

To study the private infrastructure, we use an approach to discover 694 private bridges on the Internet and a novel technique to track bridges across IP changes. We are first to measure the size of the private bridge population (35% discovered bridges are private) and to report existence of infrastructures that use private proxies to forward traffic to backend bridges or relays. We use a novel clustering approach to analyze the different infrastructures using proxies and bridges, examining its hosting and security properties. We provide an extensive discussion on the security implications of our findings.

I. Introduction

The Tor Network [35] offers protection against censorship [37], surveillance, and traffic monitoring [12], [23], by using encryption and hiding communication patterns by routing traffic through several *onion routers* (ORs). Tor keeps secret the IP addresses of a fraction of the entry ORs, called *bridges*, so that it is not trivial to block traffic destined to the Tor Network. To increase protection, bridges deploy *Pluggable Transports* (PTs) [30], which disguise Tor traffic so that it is difficult to recognize through deep packet inspection [7], [39] or active probing approaches that connect to the bridge mimicking a Tor user [41], [44].

NDSS '17, 26 February - 1 March 2017, San Diego, CA, USA
Copyright 2017 Internet Society, ISBN 1-1891562-46-0
http://dx.doi.org/10.14722/ndss.2017.23345

Research related to Tor bridges has focused on designing secure PTs [1], [21], [24], [40], [45], and proposing techniques to discover the IP address of bridges [15], [20]. However, to date there has been no security analysis of the bridge infrastructure as a whole. In this work we set out to perform the first systematic study of the Tor bridge infrastructure from a security point of view. We study both the infrastructure of *public bridges*, i.e., bridges that volunteers provide to be used by any Tor user, and *private bridges*, i.e., bridges for the exclusive use of individuals who know about their existence. While public bridges are known to the Tor Network, and report configuration and usage information to *bridge authorities*, private bridges do not report such data and thus their population and properties remain unknown. As far as we know, this is the first work that reports on Tor's private bridge infrastructure.

Public Infrastructure Analysis. To study public bridges, we leverage CollecTor [25], a public Tor service that enables access to fine-grained longitudinal data reported by public bridges (among other Tor nodes). The goal of our analysis of the public infrastructure is twofold. First, we aim at understanding whether any of the published data can harm the security of the public bridge infrastructure. We find out that usage statistics in CollecTor can be used to rank bridges by importance, e.g., by the number of clients from a specific country, or the amount of traffic carried for a particularly strong PT, which in turn allows an adversary to evaluate how well her blocking works and to identify targets. Furthermore, we find that the publication of which *OR port* a bridge uses to communicate with bridge authorities can be leveraged to optimize a scan-based search for IP addresses of public bridges and to select specific ports to scan to find a target bridge. Our findings, already reported to the Tor project, have lead CollecTor maintainers to start sanitizing the OR port data [16].

Second, we aim at measuring security-relevant properties of public bridges. We analyze the population size and its stability, finding that only 45% of public bridges carry user traffic. These bridges are long-lived and stable: their median lifetime is 4 months, and they rarely change IP address. While stability is good to increase bridge usage, it also means an adversary that discovers a bridge can block it for long periods of time without side-effects. We also observe that *default bridges*, whose IP addresses can be obtained from the Tor Browser Bundle support over 90% of bridge users, essentially defeating the very purpose of bridges. This holds in countries where censorship is known to happen such as Iran or Syria, raising the issue that a censor can at any point, e.g., in response to an event like the recent coup in Turkey, disconnect the vast majority of bridge users in the country. Finally, we analyze

PT deployment, finding that 77% of public bridges only offer the easy-to-detect vanilla Tor and another 15% offer PTs with conflicting security properties (e.g., with and without active probing protection). The latter enables an adversary to identify the bridge through the weaker PT and disable the stronger PTs by blocking the bridge's IP.

Private Infrastructure Analysis. To study private bridges, not present in CollecTor or other Tor services, we first leverage two known Tor issues [4], [38] to find their IP addresses using Internet-wide scans. This means that our study of the private infrastructure is opportunistic; fixing the Tor issues we leverage may prevent future replication of our measurements. Rather than launching our own scans, which could interfere with the Tor Network, we show how scan search engines [3], [18] can be leveraged to launch large-scale discovery of the IP addresses of bridges, with no investment in scanning infrastructure.

We follow an approach to discover bridges that uses scan search engines to find candidate IP addresses that may run an OR; connects to them to confirm their OR role; and uses CollecTor data to filter out relays and classify discovered bridges as public or private. Without launching any scan, our approach discovers 694 private bridges, and deanonymizes the IP address of 35% of public bridges with clients, and 23% of all active public bridges, in April 2016. Of all discovered bridges, 65% are public and 35% are private. We also propose a novel technique to track known bridges across IP address changes. This technique leverages additional non-Tor services (e.g., SSH) running on bridge hosts, and can be used even if the two Tor issues we leverage are fixed by the Tor project.

In the process of discovering bridges' IPs, we also uncover 645 *private proxies*, i.e., private IP addresses that forward traffic to a backend bridge or relay, and through which users can also enter the Tor Network. As far as we know, we are first to report on the existence of such private proxies. An important security implication is that discovery of a private bridge or proxy enables an adversary to flag IP addresses connecting to it as members of the owner organization, or the owner itself, and to geographically locate them.

We study the infrastructures built using proxies and bridges using a novel clustering approach to group ORs owned by the same entity based on their configuration and IP addresses. We observe 3 prevalent cluster types: (I) a line of 2 up to 178 proxies on nearby IP addresses all forwarding to the same backend OR; (II) a simplified version, with a single proxy forwarding to one backend OR; and (III) a set of bridges with no proxies, where bridges are either all public or all private. In both Type I and Type II clusters, the backend OR is typically a public bridge or a relay (but rarely a private bridge), and in 77% of these clusters the backend is in the same autonomous system (AS) as the proxies. In other words, cluster owners seem to contribute a public bridge or relay to the Tor Network, but use nearby IP addresses to run private proxies for their exclusive use. However, in general these proxies do not contribute much IP address diversity as they are hosted in the same AS and typically in nearby IP addresses. In 93% of Type III clusters, all bridges are located in the same AS, thus also raising concerns on lack of IP diversity.

II. Overview

We now present an overview of the Tor Network, focused on the components more relevant to our work. Then, in Section II-B, we describe open issues in Tor that an adversary can leverage to discover the IPs of hosts running bridges.

A. The Tor Network

The core element of the Tor Network are *Onion Routers* (ORs), also known as *relays*, which are essentially routers that forward encrypted data. A user that wants to anonymously access an Internet service runs the Tor software on its client host. This software builds a circuit of connections over three ORs, through which traffic is forwarded between the client host and Internet services. This circuit guarantees that the traffic is encrypted until it exits the circuit and that none of the relays knows both the origin and the destination of the traffic. Some of the ORs act as *directory authorities*, storing contact information for all ORs currently part of the Tor Network. Directory authorities can be queried by clients to find relays when building circuits.

Each OR is uniquely identified in the Tor Network by its *fingerprint*, which is the 20-byte SHA1 hash of its public key. ORs listen on a dedicated *OR port* for incoming connections using the *vanilla Tor* protocol [2]. The OR port is by default set to 0 [33], i.e., a freshly installed OR will not accept connections. To use the OR as a relay or bridge, the owner needs to explicitly set the OR port in the configuration file to a particular port, or to `auto` to choose a random OR port.

Bridges. Since the IP addresses of all Tor relays can be obtained at any point of time from the directory authorities, the Tor Network introduced a new OR type called *bridge*. Bridges are essentially relays that act always as first hop in a circuit, and whose IP addresses are not publicly advertised.

Pluggable Transports. An alternative way to prevent access to the Tor Network is blocking any traffic that looks like Tor communication, regardless of its destination. This is possible due to distinguishing features of vanilla Tor that are easy to detect (detailed in Section II-B). After censors started deploying deep packet inspection techniques to detect such features, the Tor Network introduced *Pluggable Transports* (PTs) [30]. A PT is just a wrapper for the Tor protocol that transforms the Tor traffic flowing between clients and bridges. Over time multiple PTs have been proposed and the most recent protocols include features such as reply protection, which guarantees that the bridge will allow connections and data transmits, only from users that previously authenticated. Pluggable Transports either imitate popular protocols (e.g., fte [5]), encapsulate Tor traffic using popular protocols like TLS (e.g., meek [8]), or are designed to look like random streams (e.g., obfs3 [28]). PTs may also implement reply protection against active probing (e.g., obfs4 [1], ScrambleSuit [42]), in which case they require users to know a shared secret before replying. A bridge can offer multiple PTs, each running on its own *PT port*.

Bridge Distribution. To use a bridge, Tor clients need to obtain its endpoint information, i.e., the IP address and port where the bridge listens for connections. Additionally, the user may need some extra information (e.g., the secret when using PTs with reply protection). Since it must not be possible for an adversary to find out the IP address of all bridges,

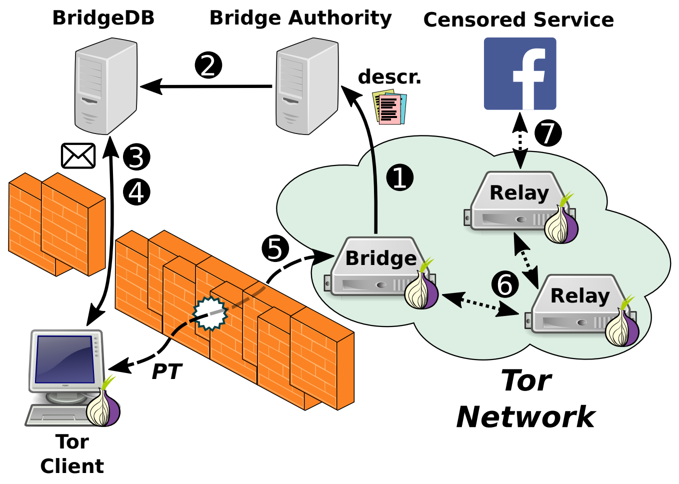

Fig. 1. Bridge distribution workflow: ❶ upon installation the bridge sends its descriptors with contact information to the Bridge Authority that ❷ assigns the bridge to BridgeDB; ❸ the user requests a bridge from BridgeDB through an uncensored channel such as email; ❹ BridgeDB sends the contact information for the bridge; ❺ the client connects to the bridge using a PT; ❻ and the bridge builds a circuit through the Tor Network. At this point, ❼ the client can communicate to the censored service through Tor.

their endpoint information needs to be carefully distributed to clients. There exist two classes of bridges: *public* and *private*. Public bridges can be used by any Tor client. They upload their endpoint information to Tor's *Bridge Authority* (or bridge directory authority), which maintains a list of available public bridges in the Tor Network. Endpoint information for public bridges is distributed to users by the *BridgeDB* service, which periodically receives it from the Bridge Authority. BridgeDB supports two different distribution channels. Users can visit its website[1] or send it an email request[2]. In both cases, users can specify the type of transport they want and whether they need a bridge that supports IPv6. Figure 1 depicts the distribution workflow for a public bridge.

The distribution algorithm adopted by BridgeDB aims at preventing the listing of a significant fraction of public bridges [17]: it only distributes a few bridges to each requesting IP address or email account, it restricts distribution to a subset of the bridge pool that changes over time, and it limits email requests to addresses from specific mail providers (Gmail, Yahoo, RiseUp).

To facilitate the use of bridges without having to go through the BridgeDB distribution channels, the Tor software ships with a list of *default bridges* for different transports. The IP addresses of these bridges are trivial to obtain, since they are hardcoded in the Tor Browser Bundle configuration files – effectively making them relays from the point of view of an adversary. Thus, these bridges can be easily blocked by adversaries. When the Tor Project detects blocking on a default bridge, the bridge is replaced by a new default.

In contrast, private bridges do not share their endpoint information with the Bridge Authority and thus are opaque to the Tor Project maintainers. Since they do not upload their descriptors on the Bridge Authority, they are not advertised to users that request a bridge from BridgeDB. Endpoint information of private bridges is distributed using private channels shared between the operator running the private bridges and the people using them.

[1]https://bridges.torproject.org/
[2]bridges@torproject.org

B. *Known Tor Issues*

In this Section we describe two known Tor open issues that we leverage to discover IP addresses of bridges.

Vanilla Tor Certificates. The vanilla Tor protocol comprises two phases. First, the client and the bridge perform a TLS handshake to agree on a shared key. Then, the two parties exchange Tor messages encrypted with that shared key. In principle, using a TLS handshake should make the vanilla Tor traffic look like TLS. In practice, the certificate chain sent by the bridge to the client during the TLS handshake is easily distinguishable, enabling to identify Tor handshakes among all TLS handshakes. In particular, the certificate chain contains a single certificate where the subject and the issuer differ and their common names have an easy to identify pattern: SubjectCN=www.[random].com; IssuerCN=www.[random].net, where [random] are base32-encoded random strings of length between 8 and 20 characters. While the certificate is changed every 2 hours, this pattern is always maintained. This issue is known by the Tor Project since at least October 2012, when a ticket was open to revise the certificates used by Tor [26]. However, the conclusion was that efforts to make vanilla Tor indistinguishable from TLS were superseded by the introduction of Pluggable Transports, and the issue was left as "wontfix" [26]. This decision should be ascribed mainly to the choice of not changing code in core parts of Tor to avoid introducing new bugs and security issues. Furthermore, developers believed that PTs were deployed widely enough for being considered the state of the art solution for users needing to bypass censorship.

Open OR port in Bridges. The second known open issue is that bridges always have an OR port open that offers vanilla Tor, even when they do not advertise vanilla Tor as a transport, but only advertise stronger PTs. Thus, bridges offering PTs will open one port per PT plus an additional one for the OR port. This issue is also known since at least November 2012, when a ticket was open for it [29]. In September 2015, the ticket priority was increased as it was considered the next major defense against bridge enumeration. But, it was also stated that the fix may require up to a month of work, as it requires changes to the Bridge Authority and BridgeDB, as well as examining multiple tools that assume bridges have an OR port.

III. Public Data Sources

We leverage two types of *publicly available* services as sources of data for analyzing the security of Tor bridges. On the one hand we use data published by the Tor project through the CollecTor service [25], which provides fine-grained configuration information and usage statistics about individual bridges and relays over time. On the other hand, we use data obtained from scan search engines, which provide information about services offered on machines connected to the Internet.

A. *CollecTor*

CollecTor is a service offered by the Tor Network that periodically collects data from Tor relays, public bridges, and other

Tor services, and makes it available online [25]. In contrast to other Tor services that provide aggregated information on the whole Tor Network (e.g., Tor Metrics [36]), CollecTor provides information at the finer granularity of individual ORs (bridges or relays). In this paper we only consider data published since July 2012, when CollecTor started to include statistics on Pluggable Transports, until April 30th 2016.

CollecTor currently publishes 16 types of files. Files for bridges and relays have the same structure, but there are some differences with respect to the published information [32]. In particular, to avoid easy identification of bridges, their sensitive data is sanitized prior to online publication. The sanitization process includes the following 5 steps: (i) replacing the bridge's fingerprint with its SHA1 hash that we call *sanitized fingerprint*, (ii) removing most cryptographic information, (iii) removing bridge contact information, (iv) removing PT ports, and (v) replacing the bridge's IP address with a format-preserving sanitized version that we call *AIP*. This AIP is of the form 10.x.x.x, where x.x.x are the 3 most significant bytes of the hash SHA256(IP || fingerprint || secret); where "secret" is a 31-byte random string that changes once per month that is used to compute the AIP of all bridges during that month.

Next, we describe the 4 files we use in this paper: bridge server descriptors, network statuses, and extra-info descriptors for studying public bridges; and network status consensuses for identifying relays and filter them from our results.

Bridge Server Descriptors. These descriptors are produced by bridges and sent to the Bridge Authority. CollecTor publishes a sanitized version containing information such as the bridge's nickname ("Unnamed" by default), sanitized fingerprint, OR port, and AIP. They may also contain contact information of the bridge operator. Unsanitized bridge server descriptors can also be obtained from bridges' themselves (if their IP is known), by connecting to their OR port. Descriptors obtained directly from the bridge contain its fingerprint (rather than its sanitized fingerprint), the real IP address (rather than the AIP), and the bridge contact information (if provided).

Bridge Network Statuses. These files are produced by the Bridge Authority and capture which public bridges are available and their current status, so they can be distributed to users by BridgeDB. While some of their data is also available in bridge server descriptors (e.g., nickname, AIP, OR port), bridge network statuses do not contain the sanitized fingerprint, but instead include the bridge uptime and the flags it has been assigned by Tor's authorities (e.g., Running, High-Bandwidth). We use bridge network statuses to measure the bridge population and its stability. In particular we use the Running flag to determine if a bridge is *active*, and thus distributed to users. This flag is assigned to a bridge if and only if the Bridge Authority was able to reach the OR port using the vanilla Tor protocol in the last 45 minutes [32].

Bridge Extra-Info Descriptors. These files are sent to the Bridge Authority by the bridges approximately once a day (every 18h by default according to [32]). These files contain the nickname and sanitized fingerprint, the PTs supported by the bridge and usage statistics (number of IPv4 and IPv6 connections, number of unique IP addresses that have connected from a country, and number of unique IP addresses that have connected using each PT). Statistics are sanitized by rounding them to the nearest multiple of 8. Still, we can use them to rank bridges according to different criteria that may be relevant for an adversary.

Network Status Consensuses. These files contain information on which relays are available in the Tor Network, their status, and their endpoint information (IP address and OR port), so that they can be chosen by clients. We use these consensus files to differentiate relays from bridges.

B. Scan Search Engines

Scan search engines index data from Internet-wide scans on a number of target ports. Each port is scanned using a popular protocol on the port, e.g., TLS on 443/tcp or SSH on 22/tcp. We use scan search engines to identify IP addresses on the Internet that serve certificates matching a specific Tor pattern. We use two different scan search engines, described below.

Shodan [18]. Scans over 200 ports using different protocols (e.g., TLS, SSH, HTTP, SMTP). When a supported protocol is identified on an IP, it indexes the service's text description (and the server certificate for TLS-based services). Among the scanned ports, 19 are scanned using a TLS handshake, which we can use for identifying IP addresses running Tor bridges (see Section IV-B). Once we identify a bridge, we query Shodan about data related to other services running in different ports of the same machine (e.g., SSH, HTTPS), in order to discover additional bridge IP addresses.

Censys [3]. Scans a smaller number of ports than Shodan using TLS (only 6) but more regularly, typically on a weekly basis. Similar to Shodan, for these 6 ports, Censys collects TLS handshake data, including the server's certificate, and it also publishes the raw scan data in addition to allowing queries on the indexed information. We download the raw data from Censys and process it locally to identify IP addresses that run ORs and have their OR port in one of the 6 scanned ports.

IV. Security Analysis Description

In this section we describe what properties we measure and the methodology used to perform those measurements. First, we introduce the measurements on public bridges performed using CollecTor data (Section IV-A). Then, we detail the approach we use to identify private bridges and proxies, (Section IV-B).

A. Public Bridges Analysis

We use the data provided by CollecTor to measure characteristics of the Tor public bridge population. Beyond understanding the demographics of the public bridges, our goal is to perform an in-depth analysis into how the fine-grained (i.e., per-bridge) publicly available data in CollecTor may impact the security of public bridges. One of the goals is to identify data that may need to be removed or to be sanitized prior to its publication. We evaluate the following 5 security-relevant properties:

(1) Bridge Population. We measure the number of bridges in the Tor Network in order to understand how large is the attack surface that an adversary needs to target for enumerating all bridges.

(2) Bridge Stability. We measure how stable bridges are in terms of lifetime and IP address changes in order to understand the vulnerability of bridges to aggressive blocking policies.

(3) Pluggable Transport Deployment. We measure the deployment of PTs over time in order to understand how long it takes to deploy a new PT and whether bridges offer multiple PTs with conflicting security properties.

(4) OR Port Distribution. We measure the frequency with which bridges use specific OR ports to evaluate how valuable this information is for an adversary that leverages the issues explained in II-B for discovering private bridges and proxies and for deanonymizing the IP address of public bridges.

(5) Bridge Importance. We rank bridges in terms of number of clients supported for different countries and PTs, showing that not all bridges are equally important. While some are rarely used, others represent vital elements in terms of the number of clients that connect to them, or the PTs they offer. Beyond improving our understanding of public bridges usage, we want to raise attention to how useful such rankings could be for an adversary. For instance, to evaluate how effective her bridge enumeration is for a given goal (e.g., "are all the bridges offering a particular transport identified?" or "are the top bridges for a country identified?"), or to target selected bridges, e.g., unblocked ones that are hard to identify because they run stronger PTs (e.g., obfs4, or ScrambleSuit).

To rank bridges, we extract usage statistics from CollecTor's extra-info descriptors. These are published periodically as introduced in Section III, which means that rankings could reflect accurate real-time information and, even though they are sanitized by rounding to multiples of 8, they still allow to order bridges in terms of number of clients they serve.

B. Private Bridges and Proxies Analysis

The goal of our analysis is to gain a better understanding of the characteristics of the private bridge infrastructure in the Tor Network, e.g., population size, configuration, and hosting. In particular for *private proxies*, which are unknown to the Tor project[3], we measure the following properties: the type of OR backend (i.e., relay, or public / private bridge) they forward traffic to, their configuration with respect to the backend (e.g., line of proxies to one backend, one proxy per backend), the ASes in which they are located, and whether the proxy ASes are the same as the backend ASes.

Since private bridges and proxies do not appear in CollecTor, studying them first requires to discover them on the Internet. We first describe our approach to identify hosts running private bridges and proxies, and then the clustering method we use to better understand their infrastructures.

Discovering Private Bridges and Proxies. We use a 5-step process to discover private bridges and proxies that leverages the open issues described in Section II-B.

Step 1 – Finding candidate IP addresses. The first step consists on performing Internet-wide scans on a selected set of ports, starting a TLS handshake on each IP:port pair, and collecting the TLS certificate when the handshake succeeds. If the certificate collected from an IP address matches the pattern associated to Tor certificates described in Section II-B, then it can be concluded that the IP address serving the certificate corresponds to a Tor OR (or a proxy to a Tor OR). To maximize the number of bridges identified with a limited scan budget, we leverage the OR port distribution that can be computed from CollecTor's data and focus on the top OR ports.

Since Internet-wide scans can be expensive to perform, and to avoid disrupting the Tor Network, we choose to substitute active scanning by queries to the Censys [3] and Shodan [18] scan search engines. Note that an adversary could similarly leverage such engines to minimize her scanning investment.

Step 2 – Filtering relays. The previous step produces a set of IP addresses running ORs (or proxies) at the time of the scan. Some of these IP addresses could correspond to Tor relays, which use the same kind of certificates as bridges. We use the Network Status Consensuses from CollecTor to classify IPs as relays. Any IP address that does not correspond to a Tor relay is a *discovered* IP address, i.e., running a Tor bridge (or proxy) at the time of the scan.

Step 3 – Verifying IP addresses. Next, our approach connects to the discovered IP address on the scanned OR port using the vanilla Tor protocol to try to download a bridge descriptor. If a descriptor is successfully downloaded, we say that the IP address is *verified*, i.e., still running a bridge (or a proxy). Furthermore, while Tor certificates are so distinct that we have not observed false positives from the regular expression used in Step 1, this step guarantees that there are no false positives since a verified IP address speaks the vanilla Tor protocol.

Step 4 – Identifying private proxies. To identify private proxies our approach compares the verified IP address from where a descriptor was collected in Step 3 with the IP address that appears in the content of the descriptor. A discrepancy between both IP addresses indicates that the verified IP address corresponds to a proxy that forwards traffic to a backend OR, to whom the descriptor belongs, running on the IP address leaked inside the descriptor. If no discrepancy is found, the verified IP address corresponds to a bridge.

Step 5 – Classifying fingerprints. A downloaded descriptor contains the bridge unsanitized fingerprint, which can be hashed to obtain the sanitized fingerprint. We then search the sanitized fingerprint in CollecTor. If found, the descriptor belongs to a public bridge, otherwise it belongs to a private bridge. For public bridges, the mapping of an IP address to a specific bridge (i.e., sanitized fingerprint) provides access to all its historical data in CollecTor.

Discovery through non-Tor Services. Once a bridge is identified, it is possible to enumerate other services offered on the host by performing a vertical scan on its IP address seeking for open ports. Those additional services may provide unique identifiers (UIDs) such as SSH keys or TLS certificates that may enable discovering other bridges from the same owners, or tracking the bridge across IP changes. The vertical scan can be replaced by querying for the IP address in Shodan, since it already scans an IP on over 200 ports with popular protocols. Once UIDs are available, periodic queries to Shodan using those UIDs can be used to find new IP addresses where the UIDs have been observed. Once a candidate IP appears, Steps 2–5 above can be applied. For public bridges the OR port from

[3]This fact was confirmed in a private conversation with Tor developers

where to try to download the descriptor can be obtained from CollecTor. For private bridges we first test the OR port of the bridge from where the UID was original found, in case it has not changed. Otherwise, Shodan is queried for open ports on the candidate IP address with a TLS certificate matching the Tor pattern. If that also fails, a vertical scan can be performed on the candidate IP (using TLS or the Tor protocol).

Clustering. To better analyze our results, we cluster public bridges, private bridges, and proxies into groups belonging to the same organization. Such clustering enables us to study the characteristics of bridge/proxy infrastructures in use. More precisely, we cluster tuples of (verifiedIP, port, descriptor) where verifiedIP and port correspond to the Internet endpoint from where the descriptor was downloaded. Our clustering uses 5 Boolean similarity features between tuples:

(1) *Same fingerprint.* Tuples with descriptors containing the same fingerprint come from the same bridge, regardless if collected from different verified IP addresses, and thus are similar.

(2) *Similar nicknames.* Nicknames are chosen by the bridge owner. Hence, we consider similar tuples with descriptors containing resembling non-generic nicknames. That is, two tuples are similar if the nicknames are identical and not generic ("Unnamed", "default", "anonymous", "ididnteditheconfig", "ididedittheconfig"), or if they have the same prefix of length 5 or more characters that is not generic ("torat", "relay", "ec2bridger"[4]), e.g., mybridge3, mybridge4.

(3) *Same contact information.* The contact information is a free-text string selected by the bridge owner that often contains an email address, but may have other content. We consider similar tuples with descriptors with identical, non-empty, contact information.

(4) *Similar verified IP address.* This feature captures that similarly configured bridges on nearby IP addresses likely belong to the same owner. Tuples whose verifiedIP is in the same /24 subnet and for which the descriptors contain identical values for 5 fields (orport, socksport, dirport, Tor version, OS) are similar.

(5) *Similar IP address in descriptor.* Tuples whose IP in the descriptor is the same or that the IP is in the same /24 subnet and for which the descriptors contain identical values for 5 fields (orport, socksport, dirport, Tor version, OS) are similar.

Two tuples with at least one of the above features returning similar are placed in the same cluster. For each cluster, we obtain statistics such as the number of fingerprints, IP addresses, private bridges, public bridges, and proxies. We also compute statistics on the hosting ASes used in the cluster and study cluster ownership based on the contact information optionally available in the descriptors of the cluster's bridges.

V. Public Bridges Analysis

In this section we analyze the data published by CollecTor about public bridges regarding features that may impact the Tor bridge infrastructure security.

[4]This generic prefix is due to the now deprecated Tor Cloud image [31].

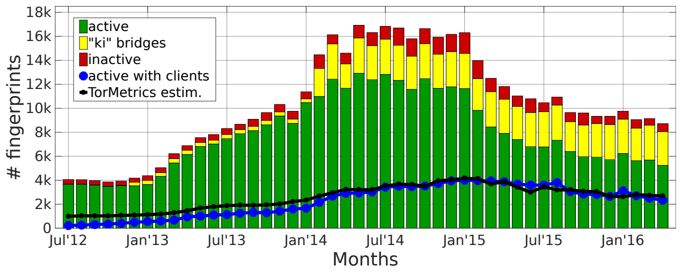

Fig. 2. Number of active (bottom bar in green), inactive (top bar in red), and *Ki* (middle bar in yellow) sanitized fingerprints over time. The two lines correspond to the monthly average number of bridges reported by TorMetrics (black line) and the number of bridges with clients (blue line with circles).

A. Bridge Population

We use CollecTor to compute the number of public bridges in the Tor Network. We uniquely identify public bridges and relays by their sanitized fingerprint, assuming that ORs change fingerprint infrequently (an assumption we validate in Section VI-A).

We split the sanitized fingerprints in CollecTor into *active* if they appear at least once with the Running flag (explained in Section III-A) in a bridge network status in a month, and *inactive* otherwise. Figure 2 shows the evolution over time of the number of active (green bar) and inactive (red bar) sanitized fingerprints in the Tor Network. The bridge population significantly varies over time: it steadily grows from 2.8K active public bridges in July 2012 up to a maximum of 12.7K in July 2014, and starts declining in January 2015 falling to 5.3K by April 2016. We have had discussions with members of the Tor project about this recent decline in bridge population, but the reason remains unclear.

The yellow middle bar represents a cluster of 3 bridges run by the same organization, that we call by their nickname, *Ki*, which change fingerprint up to once an hour (but keep their IP addresses stable, see Section VI). The *Ki* cluster produced a few dozen fingerprints in July 2012, jumped to a few hundreds in December 2012 and to a few thousands in February 2014. In March 2016, those 3 bridges are responsible for 32% of all fingerprints, corresponding to 7% of the active fingerprints and 68% of the inactive fingerprints, as most of their fingerprints do not live long enough to obtain the Running flag. After discounting those extraneous fingerprints, the number of active fingerprints in April 2016 is slightly over 5K.

The figure also shows two lines representing the monthly average number of bridges reported by TorMetrics [36] (black); and the number of active bridges with at least one client (blue with circles). These two values are very close to each other, though not the same. The blue line represents less than 50% of the active bridges in a month, indicating that more than half of the active bridges do not serve users. We examine this discrepancy in the next subsection.

We use both the number of active bridges (bottom green bar) and the number of active bridges with at least one client (blue line) as different baselines for other measurements.

B. Bridge Stability

In this section we use CollecTor to study how stable public bridges are by first measuring their lifetime (i.e., for how long

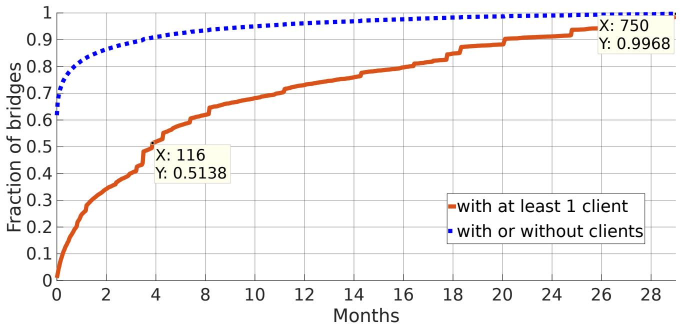

Fig. 3. CDF of bridge lifetime over time (Jul'12 - Apr'16) for all active bridges (red solid), and bridges that have had at least one client (blue dotted).

they are active) and then how often their IP address changes over their lifetime.

Bridge Lifetime. We define the lifetime of a bridge as the time window when the bridge was active, i.e., the time difference, in days, between the last and first time a descriptor for an active bridge becomes available in CollecTor. Figure 3 captures the CDF of the bridges' lifetime in our study period for all active bridges (red solid line) and for bridges with clients (blue dotted line). We see that 67% of the active bridges live for less than one day, and thus are unlikely to be used by clients. This explains the difference between the bridges with clients and the active bridges in Figure 2. However, bridges that are used by clients are quite stable. Their median lifetime is 116 days (roughly 4 months) and 25% of them live over one year.

Bridge IP Changes. Next, we evaluate how often public bridges change IP address. As explained in Section III-A, bridges' IP addresses are replaced in CollecTor with its AIP for privacy reasons. Yet, the AIP construction algorithm, which every month assigns new AIPs to bridges associated to the monthly secret, allows to compute the number of IP changes. It suffices with counting the number of AIPs assigned to a bridge's fingerprint and subtract the number of month changes in its lifetime.

Figure 4 shows the CDF for the number of IP addresses for all active bridges (blue line with circles) and for bridges with at least one client (red line with crosses). The figure shows that 67% of the active bridges have a single stable IP. This number grows for bridges with clients where 84% of bridges never change IP address, and 90% had at most one IP address change.

These results show that 55% of the bridges IPs are short-lived and thus these bridges do not carry users. On the other hand, bridges that do carry users are quite stable. They live for roughly 4 months and 84% of them never change IP address. These results have important implications for a censor: they show that current policies that remove blocks for bridge IP addresses every 25 hours [7] are extremely polite and adversaries could be performing a more aggressive blocking (up to months), without the risk of creating too many false positives.

C. Pluggable Transports Deployment

We now examine PT deployment across time and bridges. Figure 5 depicts the number of active public bridges, in logarithmic scale, offering each transport over time. It indicates that the most popular transport is vanilla Tor. Although its popularity has started to slowly decrease, it is still offered by 77% of all bridges in April 2016. The timeline shows how the deployment of obfs4 coincides with the decline of the deprecated obfs2 [34]. It also shows that after a PT is introduced, it takes between 4 months and one year to reach a stable number of 1K–2K bridges offering it. Surprisingly, deployment of different PTs does not improve beyond that stability point, an issue that we examine next.

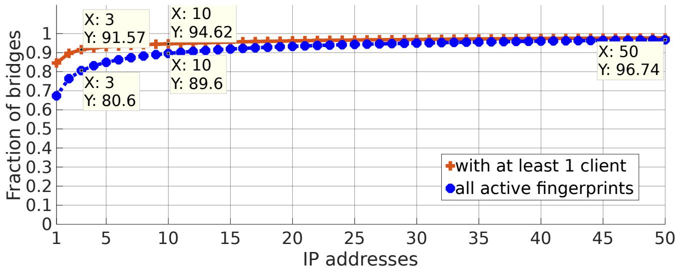

Fig. 4. CDF of number of bridge IP addresses for all active bridges (bottom) and for bridges with at least one client (top).

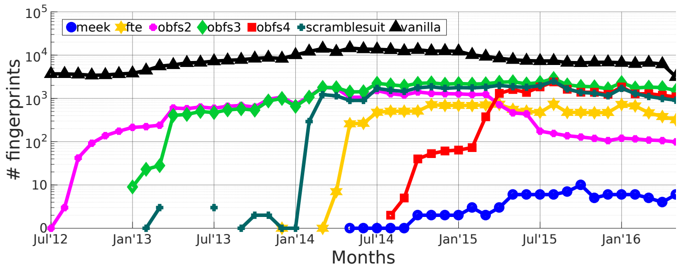

Fig. 5. Number of active fingerprints offering each transport over time (note the y-axis logarithmic scale).

vanilla	fte	obfs2	obfs3	obfs4	meek	ssuit	Bridges
✓	-	-	-	-	-	-	6,213 (77.1%)
-	-	-	✓	✓	-	✓	524 (6.5%)
-	-	-	✓	-	-	-	510 (6.3%)
-	✓	-	✓	✓	-	✓	353 (4.4%)
-	-	-	✓	✓	-	-	242 (3.0%)
-	-	-	✓	-	-	✓	129 (1.6%)
-	-	-	-	✓	-	-	117 (1.4%)
-	-	✓	✓	-	-	-	72 (0.9%)
-	-	✓	✓	✓	-	-	27 (0.3%)
-	-	-	-	-	-	✓	22 (0.3%)
-	✓	-	✓	✓	-	-	20 (0.2%)
-	✓	-	-	-	-	✓	6 (<0.1%)
-	-	✓	-	-	-	-	5 (<0.1%)
-	✓	-	-	-	-	-	4 (<0.1%)
-	-	-	-	-	✓	-	3 (<0.1%)
-	✓	-	-	✓	-	-	2 (<0.1%)

TABLE I. MOST FREQUENT TRANSPORT COMBINATIONS IN APRIL 2016. COMBINATIONS OFFERED BY A SINGLE BRIDGE OR BY INACTIVE BRIDGES ARE NOT INCLUDED.

A bridge can offer multiple transports. Table I shows the most popular transport combinations in April 2016. Rows in gray highlight that a single transport is offered. Surprisingly, 77% of the bridges only offer vanilla Tor, a transport that is trivial to identify through traffic analysis. The 1K–2K bridges offering obfs3, obfs4, and ScrambleSuit (ssuit) responds to the deployment of multiple transports on the same bridges.

The combination of PTs with different security properties raises several security concerns, since the security of the bridge is only as strong as its weakest link. First, an adversary detecting the weakest transport and blocking the IP disables also stronger transports for free, e.g., for the nearly 100 bridges that offer obfs3 or obfs4 in combination with obfs2, which is deprecated and trivial to identify through traffic analysis. Second, it allows an adversary to confirm a bridge, even in presence of transports that implement reply protection. For example, for the most popular combination obfs3+obfs4+ScrambleSuit, offered by 524 bridges, an adversary can confirm a bridge, e.g., identified through traffic analysis [39], through a vertical scan using obfs3 on the candidate IP address.

D. OR Port Distribution

In this section we use CollecTor to find the most common OR ports employed by public bridges. If an adversary knows that the majority of bridges are running on a few OR ports she can use them as targets for deanonymization via Internet-wide scanning, as described in Section IV-B.

First, we study the stability of a bridge's OR port, i.e., how often bridges change their OR port. We find that 99% of active fingerprints never change their OR port during their lifetime.

Next, we study the OR port distribution. During our observation period, bridges used 7,985 different OR ports. However, we observe that four ports (443/tcp, 8443/tcp, 444/tcp, and 9001/tcp) are chosen much more often than the rest, while all other OR ports are used only by a small subset of bridges.

Figure 6 reports the OR port usage over time. The top line corresponds to the total number of active fingerprints. The dashed (blue) line directly below corresponds to the top 4 OR ports aggregated. Each solid line below corresponds to one of the top 4 OR ports. On average, the top 4 OR ports are used by 82% of the fingerprints observed each month, although this fraction has decreased from 95% in March 2013 to 82% in April 2016. The most common port is 443. We conjecture that its popularity is due to two main reasons: it is the default HTTPS (HTTP over TLS) port, which makes it less likely for TLS-looking vanilla Tor traffic to stand out, and it is a port typically open in firewalls so it can be reached by most users. We assume that for similar reasons 8443, the alternative HTTPS port, is the second most popular port. The third most popular port, 9001, is the standard Tor port.

Port 444 is a special case since in principle is associated to the Simple Network Paging Protocol (SNPP), a not so popular protocol. However, according to CollecTor data, roughly 3K active fingerprints are using it on April 2016. The reason for this is that this OR port is used by the *Ki* bridges that change fingerprint often, as introduced in Section V-A. Those *Ki* bridges artificially inflate the usage of this OR port, a behavior that does not manifest on other OR ports.

In summary, the OR port distribution shows that an adversary could deanonymize 71% of all active public bridges by scanning 3 OR ports (443, 8443, 9001) and 82% if we consider the Top 4 with the anomalous 444. While 443 usage has declined over time, the top 4 OR ports usage closely follows the total active bridges, thus OR port diversity does not seem to be improving over time.

After we reported our findings to the Tor project, they opened a ticket to sanitize the OR port in CollecTor [16] so that our experiments cannot be replicated.

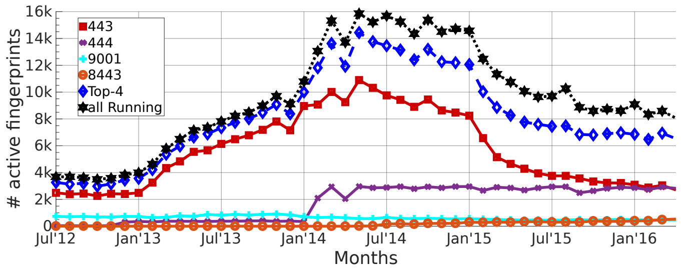

Fig. 6. Top 4 OR ports used by active public bridges over time. The top dotted line represents all active public bridges; the dashed line below corresponds the Top-4 OR ports together; and each solid line represents one OR port.

CC	Used Brid.	Clients	Top 20 (Default)	Total Default
cn	712	4,265	45.6% (44.0%) [14]	44.3% [18]
ir	941	26,479	86.6% (86.1%) [16]	86.1% [18]
sy	74	449	76.9% (68.0%) [14]	69.2% [17]
uk	943	16,723	84.1% (84.0%) [17]	84.0% [19]
us	1,496	17,911	58.7% (56.7%) [6]	56.9% [11]
All	2,213	301,009	91.71% (91.4%) [17]	91.4% [23]

TABLE II. BRIDGE IMPORTANCE PER COUNTRY (APR'16)

E. Bridge Importance

As introduced in Section III-A, not all bridges are equally important and CollecTor can be used to rank bridges according to different metrics. We now evaluate two example scenarios: ranking bridges by country usage and by PT usage. These two rankings are very relevant to, for instance, a censor that wants to know how many (or which) bridges have to be blocked to minimize the likelihood of users connecting outside the country; or to block a strong PT that it cannot be identified through traffic analysis or active probing. For our evaluation we select 5 countries: three where Internet censorship is known to occur [9]: China (cn), Iran (ir) and Syria (sy); and two where monitoring is a real threat: United Kingdom (uk) and United States (us) [12], [23].

Ranking per Country. Table II summarizes the ranking per country for April 2016. For each country it shows the country code, the number of bridges that received at least one connection from users located inside the country, the average number of clients per day served by those bridges, the percentage of these clients served by the Top 20 bridges, with the fraction served by default bridges[5] in parentheses, and the total traffic carried by default bridges. For the last two columns the number of default bridges is shown in square brackets. The *All* row at the bottom corresponds to statistics for all bridges in CollecTor regardless of user location.

The Top 20 bridge statistics show that a few bridges handle the majority of clients. The Top 20 support more than 91% of traffic for the full Tor Network, and more than 75% for all countries except China and the US, where clients are better spread among available bridges. The Top 20 is dominated by default bridges. In fact, the Top 14 bridges in the full Tor Network, and also in all countries except the US, are default

[5]We use the default bridges available in the Tor Browser 5.5.5 distribution.

PT	Used Brid.	Clients	Top 20 (Default)	Total Default
vanilla	1,967	14,939	5.6% (0.0%) [0]	1.2% [21]
obfs2	13	158	100.0% (25.8%) [1]	25.8% [1]
obfs3	898	63,088	92.0% (90.8%) [4]	90.8% [4]
obfs4	792	204,095	95.4% (94.7%) [11]	94.7% [11]
ssuit	467	4,483	52.4% (46.3%) [1]	46.3% [1]
meek	4	22,685	100.0% (∼100%) [3]	∼100% [3]

TABLE III. BRIDGE IMPORTANCE PER PT (APR'16).

RK	Port	Clients	BRs	Ranking per Country				
		(%)	[Default]	cn	ir	sy	uk	us
1	6666	23.805%	1 [1]	2	5	6	1	1
2	42506	14.096%	1 [1]	6	3	4	3	-
3	60906	13.877%	1 [1]	7	4	3	2	-
4	63848	13.730%	2 [2]	5	6	5	4	4
5	44445	9.485%	1 [1]	8	2	2	5	2
6	8008	7.173%	1 [1]	4	54	-	6	-
7	29001	5.027%	2 [1]	10	1	1	7	3
8	9002	2.827%	2 [1]	1	7	8	8	-
9	1512	1.206%	1 [1]	3	8	14	9	125
10	9001	0.263%	309 [6]	19	9	7	10	5
11	29309	0.045%	1 [0]	36	10	-	42	10
12	27134	0.041%	1 [0]	15	13	18	12	16
13	20506	0.040%	1 [0]	59	19	19	11	7
14	12497	0.040%	1 [0]	57	14	-	42	9
15	59760	0.039%	1 [0]	18	19	-	33	11
16	60841	0.039%	1 [0]	49	15	-	50	16
17	53885	0.038%	1 [0]	15	36	-	50	14
18	14769	0.035%	1 [0]	38	61	-	11	6
19	34678	0.033%	1 [0]	37	12	-	66	8
20	19924	0.032%	1 [0]	12	19	-	19	14

TABLE IV. OR PORT RANKING FOR MOST USED BRIDGES (APR'16).

bridges. Thus, we can conclude that default bridges carry the vast majority of clients in the Tor Network.

This dominant choice of default bridges, which are trivial to deanonymize through the configuration files shipped in the Tor Browser Bundle, means that in the majority of cases clients route their traffic through known bridges, thus defeating the very purpose of bridges.

Ranking per PT. Table III summarizes the ranking per PT for April 2016. Its structure is the same as that for Table II. The results indicate that while vanilla Tor is the most widely deployed transport, most clients opt for more recent PTs such as obfs4, obfs3, and meek. ScrambleSuit (ssuit) is not very popular and, as expected [34], obfs2 has almost completely stopped being used.

We observe that the Top 20 bridges handle almost the entire user base for obfs4, obfs3, and meek. The vast majority (90%–100%) of clients using these 3 transports are carried by default bridges. The Top 20 bridges do not dominate vanilla Tor and ScrambleSuit, likely related to no default bridge being marked as offering vanilla Tor in the Tor client configuration files and only one offering ScrambleSuit. Still, the lone default bridge offering ScrambleSuit carries over 46% of the clients using that transport.

OR Port Distribution per Country. In Section V-D we have studied the global ranking of OR ports and have shown that the top 3 OR ports are used by 71% of all public bridges. Here, we analyze whether the top bridges by number of clients also follow that distribution.

Table IV contains the Top 20 OR ports by percentage of clients served in the full Tor Network. For each OR port it reports the rank, the percentage of clients supported by bridges using that OR port, the number of bridges using that OR port (and how many are default bridges in square brackets), and the ranking of the OR port when only considering connections from a particular country.

The table shows that the choice of OR port among popular bridges does not resemble the distribution of OR ports in the overall bridge population. The Top 6 OR ports correspond exclusively to default bridges. The first OR ports in the Top 10 global distribution appear at rank 8 (9002) and rank 10 (9001). In both cases, the majority of the clients served on those ports correspond to default bridges, e.g., for port 9001 the percentage of clients served by the 303 non-default bridges is negligible. From that point, we see that the most popular bridges run in random high ports. These results suggest that owners of non-default popular bridges are careful to set the OR port selection to random, but those of less popular bridges are not as careful in this respect.

The ranking per country shows that, even though in general default bridges dominate in all countries, the popular non-default bridges can significantly vary across countries. For example, in Syria the globally popular bridges have little overlap with the popular ones in that country. This suggests that a state-level adversary can use CollecTor to compute a ranking of, currently unblocked, OR ports that she should target next through Internet-wide scanning to maximize the blocked population in her country.

Another observation (not shown in Table IV) is that all Top 20 non-default bridges incur in the problem flagged in Section V-C of offering multiple PTs with different security properties.

VI. PRIVATE BRIDGES AND PROXIES ANALYSIS

In this section we analyze private bridges and proxies in the Tor Network. Section VI-A presents the results of applying the bridge discovery approach described in Section IV-B on the Tor Network during April 2016. Then, in Section VI-B we analyze the discovered bridge/proxy infrastructures.

A. Discovering Private Bridges & Proxies

Since we do not run our own scans, we can only apply the bridge discovery to 7 out of the top 10 OR ports, which are scanned by either Censys or Shodan using TLS in April 2016. Table V summarizes our results. For each OR port we report: the number of scans available on that port in April 2016, the source of the scanning data (either Censys or Shodan), the first and last scan dates, the date when verification started, the number of discovered bridge IP addresses, i.e., those where scan data shows a certificate matching the Tor pattern and that are not relays, the number of verified IP addresses (and fingerprints in parentheses) from which we were able to download a bridge descriptor, and the split of verified IPs (and fingerprints) into public bridges, private bridges, and private proxies. Note that we distinguish public and private bridges through fingerprints, but proxies by IP address, as proxies are

Port	**Scans**	**Source**	**Scan Dates**	**Verif. Date**	**Disc.**	**Verified**	**Public**	**Private**	**Proxy**
443	9	Censys	04/04–04/28	04/08	2,448	1,315 (1,122)	897 (860)	263 (262)	164
993	2	Censys	04/20–04/27	04/21	19	16 (13)	11 (11)	3 (2)	2
995	3	Censys	04/15–04/29	04/23	14	14 (13)	10 (10)	3 (3)	1
444	1	Shodan	04/19–04/19	04/19	14	12 (101)	8 (97)	1 (4)	4
8443	1	Shodan	04/21–04/21	04/22	191	156 (149)	148 (148)	1 (1)	7
9001	1	Shodan	04/17–04/17	04/18	2,001	1047 (587)	165 (166)	415 (421)	468
9002	1	Shodan	04/23–04/23	04/23	23	19 (5)	1 (1)	4 (4)	14
All	17	All	04/04–04/29	04/08	4,684	2,554 (1,986)	1,239 (1,292)	684 (694)	645

TABLE V. BRIDGE DISCOVERY IN APRIL 2016

not Tor ORs and have no fingerprint. The last row shows the aggregate results for all ports.

Overall, we discover 694 private bridges and 645 private proxies, which do not appear in CollecTor. Additionally, we deanonymize the IP address of 1,292 public bridges. According to CollecTor data, these correspond to 35% of public bridges with clients (23% of all active public bridges) in April 2016, excluding *Ki* bridges. On the 7 OR ports examined, the bridge population comprises 65% public and 35% private bridges, i.e., one in three bridges is private.

There exist several reasons why we do not deanonymize a larger fraction of public bridges in CollecTor. First, we can only deanonymize bridges on the 7 OR ports scanned by Censys or Shodan. Second, for most ports we only have one scan in the second half of the month. Hence, we cannot discover short-lived bridges active only in the first half of the month. Furthermore, we have shown that 55% of active bridges are short-lived; thus, while they may appear as candidate IPs, we may not be able to confirm them because they no longer live when we try to download a descriptor from them.

We note that the aggregated results differ slightly from the sum of all rows for two reasons. First, a few proxies have more than one OR port open and thus their IPs are counted in more than one row. In addition, a few IP addresses were observed hosting a proxy at some point in the month and hosting a bridge at other times. Also, note that the number of verified fingerprints for port 444 is significantly larger than the number of verified IPs. This is because 6 of these IPs (3 proxies and 3 public bridges) belong to the *Ki* cluster that changes fingerprint periodically.

Discovery through non-Tor Services. Next, we evaluate if additional services running on bridge hosts can be leveraged to track bridges across IP address changes. For this, we assume an adversary discovered a bridge by any means, (e.g., the approach in Section VI-B, querying BridgeDB, or adding a middle OR in the Tor Network [15]), but the open OR port issue we leverage has been solved by the Tor project.

First, we measure the percentage of bridges in Table V with additional ports open (beyond the OR port). For this, we query each verified IP address at Shodan. Overall, 621 (24%) of the 2,554 verified IPs offer at least one additional service (beyond the OR port) and 10% more than one. In total, we observe 101 additional ports. These numbers indicate that it is not uncommon to run other services on a Tor bridge. The most common additional services are SSH on ports 22 and 2222, Web services on ports 80 and 443, and RPC port mapper on 111. As unique identifiers (UIDs), we use SSH keys on ports 22 and 2222 and certificate serial numbers on 443.

Scanning those UIDs in Shodan on May 18 provides us with 2,248 candidate IPs, of which only 248 return a descriptor. After filtering out relays and already known IPs, we found 9 new bridge IP addresses that were not observed in April 2016. For example, one is located in Amazon EC2 and corresponds to a bridge that was running on another IP in April 2016, but likely changed IP because the EC2 instance was restarted, as EC2 assigns VM IP addresses from a shared pool.

Fingerprint Stability Validation. The discovery of bridges, which provides us with access to their unsanitized descriptors, allows us to validate the assumption that OR fingerprints rarely change, and thus are indeed good bridge identifiers. We periodically (roughly once a day until June 3rd, 2016) try to download a descriptor from verified IP addresses. Then, we measure the frequency of fingerprint changes in the descriptors for bridge (i.e., non-proxy) IPs found in April 2016.

Overall, 94.1% of the bridge IP addresses did not change fingerprint, 5.5% changed fingerprint once, and 0.4% changed fingerprint multiple times. The bridges with multiple fingerprint changes include the 3 *Ki* bridges, which present a different fingerprint every time we connect to them (on a closer look we find that they change fingerprint roughly every hour). Furthermore, we observe that over 70% of the IP addresses with fingerprint changes belong to 2 clusters of private bridges each using multiple nearby IP addresses. These IPs change fingerprint on the same dates, so it is possible that bridges in each cluster were reassigned IP addresses on those dates.

These numbers confirm that the vast majority of bridges do not change fingerprint over time. Thus, bridge fingerprints (and sanitized fingerprints in CollecTor) can be used as identifiers.

Hosting. The discovered proxies are hosted in 166 ASes, the private bridges in 139, and the public bridges in 385. The top 10 ASes for both bridges and proxies are dominated by the ASes used by the top clusters we find in the next Section. For bridges, the top 10 ASes comprise: 5 popular cloud hosting providers, 2 broadband residential ISPs, and 3 large ISPs that provide multiple services. While overall there seems to be enough IP diversity among the bridge population, in the next section we show that individual clusters exhibit less diversity.

Contact Information. Bridge descriptors may contain contact information that could reveal ownership. We find 267 email addresses in the descriptors collected from the 1,986 public and private bridges. Of those, 69 have a domain name from a public email service provider (e.g., Gmail, Yahoo), and thus cannot be easily used to establish ownership. The other 198 email addresses contain 191 domain names, of which 175 return a valid mail server IP address through a DNS MX query, the rest we discard as invalid. Those 175 domains appear

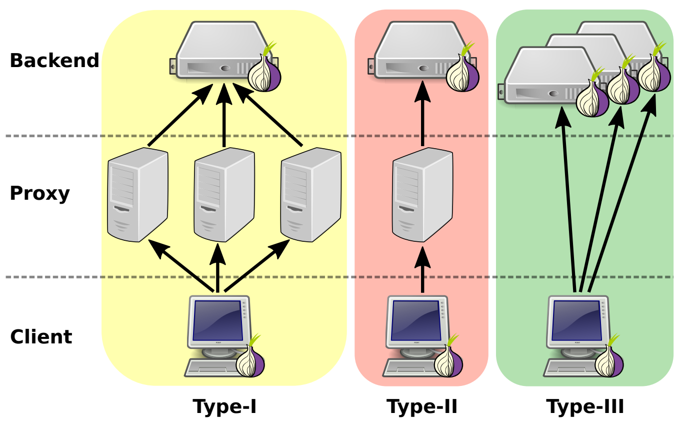

Fig. 7. The three most common cluster types.

#	Type	IPs All	IPs Proxy	Bridges Pub.	Bridges Priv.	ASNs	Contact
1	I public	179	178	1	0	1,1,✗	✓
2	III private	164	0	0	164	0,1,✗	✓
3	I relay	72	71	0	0	1,1,✓	-
4	III private	63	0	0	63	0,1,✗	-
5	III public	53	0	53	0	0,16,✗	✓

TABLE VI. STATISTICS ON TOP 5 CLUSTERS.

in descriptors from 307 bridges: 187 public and 120 private. This indicates that, contrary to what could be expected, private bridges often provide contact information that may reveal the organizations behind the bridge. This enables an adversary to identify Tor clients connecting to the bridge as members of that organization.

B. Bridge/Proxy Infrastructures

We cluster bridges and proxies as described in Section IV-B to better understand how they are used. Out of 41,359 tuples (verifiedIP, port, descriptor), collected by connecting periodically to the 2,554 verified IP addresses, we obtain 1,343 clusters, of which 75% are singletons, i.e., contain a single bridge and no proxies.

We identify 5 cluster types, as well as mixed clusters with subclusters of those types. The 3 most common cluster types are shown in Figure 7. Type I corresponds to a line of proxies (from 2 up to 178) that all forward to the same backend. Type II is a Type I cluster with a single proxy. For both types, the backend can be a private bridge, a public bridge, or a relay. Type III is a cluster of multiple bridges belonging to the same owner, without proxies. Type III clusters can have only public bridges or only private bridges.

Not shown in Figure 7 are the rarer Type IV and Type V. Type IV corresponds to the same bridge running on multiple IPs *at the same point in time*. This differs from a bridge changing IP over time where we would observe the same fingerprint at different IPs on different days. This cluster type could be a result of cloning an image with an installed bridge on multiple VMs. Type V corresponds to a proxy load-balancing across multiple backend IPs. In most cases, the backend IPs host the same bridge (fingerprint), which runs on a Windows host. This is similar to a fast-flux network where residential hosts run bridges, and the proxy provides IP stability to access them.

We develop a set of rules to automatically classify non-singleton clusters. The cluster classification outputs: 47 clusters as Type I (37 relay, 7 public, 3 private); 138 as Type II (94 relay, 29 public, 15 private); 88 as Type III (69 public, 19 private); 43 as Type IV; 10 as Type V; and 16 as mixed clusters containing at least two subclusters of other types.

Type I clusters most often have proxies forwarding to a relay (37 clusters) or a public bridge (7 clusters). A similar trend applies to Type II clusters. In all Type I clusters all proxies in the cluster are hosted in the same AS, and are often located in nearby and even consecutive IP addresses. In 77% of Type I and Type II clusters, the proxies and the backend are in the same AS. The cases where proxies and backends are in different ASes could indicate organizations free-riding on public ORs to perhaps avoid configuring their own bridge. We observe only 3 Type I clusters with private backend indicating that, since proxy IPs are already private, there is no benefit on a private backend.

In Type I and Type II clusters, proxies seem to be used by owners of public bridges and relays to keep a few IPs (only one for Type II) private for their own use. Those proxies do not provide much IP diversity; once a proxy is known, an adversary could scan the nearby IP addresses to find other proxies and the backend. Furthermore, when the backend is a relay, an adversary could scan IP ranges hosting relays to try to find proxies forwarding to it. It is also important to note that when the backend is a relay only vanilla Tor is supported, as relays do not run PTs.

Type III clusters most often comprise public bridges (69 clusters) but can also have private bridges (19 clusters). In 93% of Type III clusters all bridges are in the same AS. This indicates that once an owner has established a relationship with a hosting provider, it is easier to install multiple bridges on that provider. Still, there are a few Type III clusters with good diversity. For example, the cluster at Rank 5 has 53 public bridges on 16 ASes. We observe that some Type III public clusters belong to organizations that also contribute relays to the Tor Network. Another public cluster belongs to a computer security company.

The mixed clusters show that our clustering can capture multiple infrastructures from the same owner. An example of a mixed cluster is the `Ki` cluster, which comprises a Type I subcluster with 2 proxies forwarding to a public bridge, another Type II subcluster with one proxy forwarding to one public bridge, and a singleton subcluster with a public bridge. Both bridges and proxies in the cluster keep their IP address stable, but bridges change fingerprint hourly.

We conclude that proxies are not generally used for load-balancing across multiple backend ORs as there are only 10 Type V clusters compared to 185 Type I and Type II clusters. Proxies do not seem to be used for OR port diversity either. We observe only 13 proxy IPs forwarding on multiple ports. Of those, 10 belong to 2 clusters where the same backend IP runs two different bridges on different ports and a proxy port forwards always to the same backend OR port.

Top Clusters. Table VI provides details for the Top 5 clusters. For each cluster, it shows the rank by number of IP addresses;

the type; the total number of IP addresses; the number of proxy IP addresses; the number of public and private bridges (by fingerprint); the hosting as a (x,y,flag) tuple with number of autonomous system numbers (ASNs) for proxy IPs (x), bridge IPs (y), and whether those ASNs are the same (✓) or not (✗); and if the bridges in the cluster contain contact information.

VII. Security Discussion of Findings

In this section we recapitulate the findings described throughout the paper and discuss their implication with respect to the security and privacy of Tor's bridge population.

A. *Security Implications of Scan Search Engines*

While the use of Internet-wide scanning for discovering bridges was known [4], [38], our work illustrates how the availability of scan search engines, which index data collected from Internet-wide scans, greatly lowers the attack cost in terms of scanning infrastructure. Thus, the security of systems that require hiding components, e.g., Tor bridges, should be designed with this threat in mind.

By leveraging search engines, an adversary that follows our approach can discover, with no investment in scanning infrastructure, 35% of public bridges with clients, 23% of active public bridges, and hundreds of private bridges and proxies. The discovered public bridges support 95% of all clients and include 90% of the bridges offering obfs4 and obfs3. In contrast, other bridge enumeration attacks, e.g., using a middle node or performing clever queries to the Bridge Authority [15], achieve lower coverage, are much slower, and may require setting up an OR. Furthermore, we have shown a novel technique to leverage scan search engines to discover Tor bridges by examining non-Tor services running on bridge hosts. Those services may provide unique identifiers (e.g., SSH keys) that may reveal other bridges from the same owners and enable tracking a bridge across IP changes.

From the perspective of resource usage, our whole processing runs on a single host, with no demanding requirements in terms of CPU and memory. The main bottleneck is the time needed to download the Censys scan, which typically exceeds 100 GB and takes around 24 hours. The reason we download the raw scan data, rather than querying Censys directly, is that the Censys administrators seem to be filtering Tor certificates from the indexed (but not the raw) data. Downloading the smaller CollecTor files can be done in parallel with the Censys download. Beyond the data download, it takes 8-12 hours to process the Censys data, query Shodan, and connect to each discovered IP using vanilla Tor. Thus, our processing currently incurs a delay of one and a half days between a bridge is scanned and we connect to it.

We note that current Internet-wide scanning approaches focus on IPv4 addresses. Bridges running exclusively on IPv6 addresses are difficult to identify through such scanning due to limited scan coverage. While such bridges can only be accessed by clients supporting IPv6, recent OSes support IPv6 off-the-shelf. Thus, a possible defense against our techniques is to transition a fraction of bridges to IPv6.

Closing the OR Port. Having the OR port open in all bridges enables bridge enumeration through Internet-wide scanning. Closing the OR port has been queued for fixing since November 2012. While its priority was increased in September 2015 [29], the fix has not happened yet, as it requires changes to the Bridge Authority, BridgeDB, and multiple other tools that assume bridges have an OR port. Our quantification of the impact of this bug highlights the importance of quickly fixing vulnerabilities. We hope our work will provide a push to fix this bug and possibly to find an entity that sponsors the fix.

While the bug is fixed, possible stopgap fixes include allowing only known IPs (e.g., Bridge Authority) to connect to the OR port and improving OR port diversity. The latter is critical because closing the OR port by default does not solve the problem that it would remain open for the 77% bridges offering vanilla Tor, which would still be vulnerable to scanning attacks. Non-default important bridges, i.e., carrying a significant number of users, seem to already be randomizing the OR port by setting it to `auto` in their configuration, likely due to the expertise of their owners. This forces an attacker to scan many ports before finding a significant fraction of bridges. However, we have shown that 71% of public bridges use 3 OR ports. Thus, it is critical to educate less expert users on the importance of setting the OR port to `auto` in their bridge configuration. Bridge configuration tutorials that still use a fixed OR port, e.g. [27], should be fixed.

We must stress that closing the OR port does not eliminate the threat of discovering bridges through additional non-Tor services coexisting on a bridge's host, nor other security risks uncovered by our analysis that we discuss in the following sections.

B. *Security Implications of CollecTor Data*

While collecting fine-grained data about the Tor Network is fundamental for the Tor project maintainers, making it publicly available through services like CollecTor requires an analysis of the benefit versus privacy trade-off to understand the risks. One of our goals was analyzing if the fine-grained, per-bridge, data provided by CollecTor, as opposed to aggregate statistics provided by Tor Metrics, could be harmful to the security of public bridges. Here we discuss our findings.

OR Ports. The most concerning data in CollecTor we found is the availability of OR ports. Knowing the OR port of all public bridges helps to optimize the discovery of bridges by focusing Internet-wide scans on the most popular OR ports, which are used by the overwhelming majority of bridges. Furthermore, as discussed in Section V-E, an adversary can search for the OR port of a bridge of interest (e.g., one supporting most users in a censored country or providing strong PTs) and then perform an Internet-wide scan on that OR port to deanonymize it. After we sent a draft of this work to the Tor project, CollecTor has started sanitizing the OR port in a similar way as they anonymize bridge IPs [16]. Such sanitization prevents mapping an anonymized OR port in CollecTor to a real port, preventing the two issues above.

Fingerprints. A second piece of data extremely valuable for our work was the availability of the bridge sanitized fingerprint. It allowed us to link information from different CollecTor files and gather longitudinal information on individual bridges. Having a unique bridge identifier is fundamental to the granularity of CollecTor. Thus, it does not seem easy to remove them and instead raises the question of whether CollecTor should be a service only available to Tor maintainers. Another issue is that once the fingerprint of a public bridge is known, it can be hashed to find the specific bridge in CollecTor enabling near-real-time access to bridge's data (e.g., PTs offered, IP changes), as well as longitudinal statistics on the whole bridge lifetime. This could be addressed by adding a secret to the fingerprint sanitization. However, this would prevent bridge owners to check CollecTor statistics on their own bridges.

Usage Statistics. Rounding usage counters is a simple method to protect individual users while enabling statistics collection. Recent developments on privacy-preserving collection of statistics [6], [13] improve this protection and enable a wider range of statistics collection. Yet, these methods are not sufficient to prevent an adversary from ranking bridges according to their usage. We have shown that such rankings have two security implications. First, they enable an adversary to evaluate how successful is her blocking, not only at the global Tor Network level, but more worryingly for specific countries and PTs. Second, they allow an adversary to identify valuable (yet unblocked) bridges to target. Therefore, we believe that further research on privacy-preserving publishing of aggregated statistics is needed.

C. Security Implications of Bridge Properties

In this Section we discuss implications of the bridge properties we learned through the analysis of CollecTor data.

Bridge Stability. We found that public bridges can be coarsely classified in 55% volatile and 45% stable, where only stable bridges carry client traffic. Volatile bridges are short-lived and may be due to bridges installed on machines not always connected to the Internet, or by users testing how to run a bridge. Stable bridges are long-lived (median lifetime of 4 months) and rarely change IP address. Stability means once a user obtains the bridge information from BridgeDB it can use it for a long time. On the other hand, it also implies that current adversary blocking policies, e.g., the GFC removing blocks for bridge IP addresses every 25 hours [7], are extremely polite. Once an adversary finds a stable bridge, it can perform more aggressive blocking (up to months) or adaptive, by reconnecting to a bridge every day to check it is still active and the block should be renewed.

Use of Default Bridges. Our study of bridge importance reveals that default bridges carry over 90% of bridge users. Default bridges enable out-of-the-box use of Tor software, without the need to request bridges from BridgeDB. While censors may not be continuously blocking default bridges (otherwise they would not carry clients on censored countries), their massive usage enables easy disconnection of the bridges user base in response to events. Our measurements show that such blocking would disconnect nearly 90% of bridge clients in countries like Iran and Syria. Additionally, the fact that Tor users are educated to use the software out-of-the-box casts doubts about their ability to find alternative bridges when such blocking happens.

PT Deployment. We observe that 77% of public bridges only offer vanilla Tor and another 15% mix PTs with conflicting security properties (e.g., with and without reply protection), reducing the protection to that of the least safe transport. In general, the goal should be that bridges do not offer weak transports (e.g., vanilla) or deprecated ones (e.g., obfs2), but only PTs considered safe and without conflicting properties. The current PT deployment strategy in which bridge owners decide independently which PTs to offer from the complete pool of PTs is not optimal. There is a need for a faster way to remove PTs known to be unsafe (e.g., vanilla, obfs2). This could be achieved by adding automatic updates to Tor, enabling centralized decision on which PTs should no longer be offered and faster distribution of updates to disable them. In general, automatic updates would more quickly close the vulnerability window for any already fixed security issue. Since adding automatic updates may take time, the Tor software could in the mean time be configured to offer the strongest PT (e.g., obfs4) by default and to warn the user if two transports with conflicting security properties are about to be offered.

D. Security Implications of Uncovering Private Infrastructure

Discovery of private bridges and proxies is arguably more worrisome than deanonymization of the IP address of public bridges. Their discovery allows an adversary to learn that IP addresses connecting to them correspond to Tor users that are members of the owner organization, and to use their IPs to geographically locate the users. This is particularly dangerous as private bridges and proxies may be run precisely by organizations trying to avoid such identification. One positive aspect is that since private bridges do not report to the Bridge Authority, nor are explored by any Tor-related service, it may be much easier to disable their OR port. This would prevent private bridges from appearing in scan search engines, and if they exclusively use strong PTs (e.g., obfs4), would thwart attempts of scanning to find them.

We have uncovered that multiple organizations are using private proxies as cheap replacements for private bridges. However, proxies are always in the same AS, which in 77% type I and II clusters is also the same AS hosting the backend OR. Once a proxy is discovered, it is possible to scan nearby IP addresses to find other proxies and the backend. In the case of backend relays, whose IP addresses are known, an adversary can perform localized scans around relay IP addresses to locate proxies. In those cases, using as backend a public bridge with a random OR port is preferable to a relay.

Proxies also have non-security implications for the Tor Network. First, they add an extra hop in addition to the 3-hop Tor circuit, which if the proxy lays across the Internet from the bridge increases the, already high, latency of Tor circuits. Second, proxies affect the usage statistics in CollecTor as connections from multiple clients, potentially in different countries, are all counted as connections from the proxy IP address by the backend OR.

VIII. Related Work

The academic community has paid quite some attention to Tor bridges A first line of research related to this work deals with the proposal of PTs to avoid traffic analysis attacks [5], [21], [40], [45], and their detection [39]. The latter work studies the detectability of five popular PTs. Their results show that a determined adversary can reliably detect communications with bridges with a low false positive rate.

A second line of work arises in response to censors using active probing to confirm that suspicious nodes are bridges [7]. This work also shows that the censors' systems operate in real time, are able to detect servers using five circumvention protocols and are regularly updated. Houmansadr et al. [10] show that PTs that mimic other protocols, e.g., [21], [40], are particularly sensitive to active probing since they only mimic part of the communication. Defenses against active probing are based on PTs that only reply upon being proven that the client knows a long-lived secret [1], [45] or a short-lived key [24].

A third relevant research line is dedicated to the discovery of bridges. McLachlan and Hopper [20] showed that it is possible to deanonymize bridge operators when bridges run in clients. In case bridges are dedicated servers, Ling et al [15] provide both active attacks where the adversary directly interacts with the bridge distributor, and passive attacks where she sets up a relay and enumerates bridges (i.e., non relay nodes) that connect to it. The work on Internet-wide scanning by Durumeric et al. [4] showed that the pattern in Tor certificates could be used to identify bridges, a property that we leverage to realize our measurement of the Tor bridge infrastructure. We show how to optimize enumeration by leveraging data in CollecTor and the availability of scan search engines to minimize investment in scanning infrastructure. We also present an alternative discovery technique based on non-Tor services bridges may also run, which is related to works leveraging leaks on scan search repositories to deanonymize hidden services [19].

A fourth research line proposes defenses based on *decoy routing*, where modified routers can be signaled to act as circumvention proxies [11], [14], [22]. These solutions increase the cost of blocking at the expense of modifying the routing infrastructure, an orthogonal problem to the one in this paper where we study the security of currently deployed defenses.

Finally, Winter et al. [43] propose a tool to identify sybils in the Tor Network, i.e., relays owned by the same group, by studying configuration and uptime similarity. Our clustering has a similar goal as their nearest neighbor ranking and can be applied to relay descriptors as well, with the advantage that it does not require an input OR to compare with.

IX. Ethical Considerations

Our measurements only leverage known limitations of the Tor Network, and only use leaks present in publicly accessible repositories such as Shodan, Censys or CollecTor. We purposefully avoid adding relays or bridges into the Tor Network, as well as exploiting any software vulnerability. We have no access to any traffic that is not ours, and hence we can not threaten the privacy of any Tor user. However, the data we collect contains the IP addresses and contact information of public and private bridges that must be kept private to preserve the security provided by the Tor Network. Thus, we do not disclose any bridge/proxy IP addresses, nor any personal information we may learn about its owners, but only provide aggregate data to illustrate important steps and findings.

This work has been approved by ethics review board of our institution, which has mandated that due to its sensitive nature the data must be protected with diligence, must not be disclosed to third parties, and must be deleted when the paper is published. We sent a copy of the submitted draft to the Tor project. They have already started taking measures [16] to prevent bridge targeting based on CollecTor public information.

X. Conclusion

In this work we provide the first systematic security analysis of the Tor bridge infrastructure. Our opportunistic measurements, made possible by taking advantage of two known Tor issues, allow us to discover thousands of bridges which we have used to gain understanding about the security properties of the bridge population. In particular we uncover the use of private proxy-based infrastructures likely to obtain IP diversity to access the Tor Network. We also study the impact on security of publicly available information such as that provided by CollecTor or scan search engines. Our results have implications for the Tor project, since they indicate that the two issues we leveraged need to be solved as soon as possible, and that the information offered by CollecTor may need to be reduced; but also beyond, since we confirm that the information made available by public scan search engines should be taken into account when designing covert services.

Acknowledgments

The authors would like to thank the Tor Project for their feedback. In particular, we are very thankful to Karsten Loesing for promptly launching a fix to anonymize OR ports in CollecTor and to Steven Murdoch for useful discussions about bridges' OR ports. We would also like to thank John Matherly and the Shodan project for their help.

This research was partially supported by the Regional Government of Madrid through the N-GREENS Software-CM S2013/ICE-2731 project, by the Spanish Government through the DEDETIS Grant TIN2015-7013-R, and by the European Commission through the H2020-ICT-2015 NEXTLEAP project (GA 688722). All opinions, findings and conclusions, or recommendations expressed herein are those of the authors and do not necessarily reflect the views of the sponsors.

References

[1] Y. Angel. obfs4 (the obfourscator), 2014. https://github.com/Yawning/obfs4/blob/master/doc/obfs4-spec.txt.

[2] R. Dingledine and N. Mathewson. Tor protocol specification. https://gitweb.torproject.org/torspec.git/tree/tor-spec.txt.

[3] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman. A Search Engine Backed by Internet-Wide Scanning. In *ACM Conference on Computer and Communications Security*, 2015.

[4] Z. Durumeric, E. Wustrow, and J. A. Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *USENIX Security Symposium*, 2013.

[5] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton. Protocol Misidentification Made Easy with Format-Transforming Encryption. In *ACM Conference on Computer and Communications Security*, 2013.

[6] T. Elahi, G. Danezis, and I. Goldberg. PrivEx: Private collection of traffic statistics for anonymous communication networks. In *ACM Conference on Computer and Communications Security*, 2014.

[7] R. Ensafi, D. Fifield, P. Winter, N. Feamster, N. Weaver, and V. Paxson. Examining How the Great Firewall Discovers Hidden Circumvention Servers. In *ACM Internet Measurement Conference*, 2015.

[8] D. Fifield, C. Lan, R. Hynes, P. Wegmann, and V. Paxson. Blocking-Resistant Communication through Domain Fronting. In *Privacy Enhancing Technologies Symposium*, 2015.

[9] Freedom House. Freedom on the Net, 2015. https://freedomhouse.org/report/freedom-net/freedom-net-2015.

[10] A. Houmansadr, C. Brubaker, and V. Shmatikov. The Parrot Is Dead: Observing Unobservable Network Communications. In *IEEE Symposium on Security and Privacy*, 2013.

[11] A. Houmansadr, E. L. Wong, and V. Shmatikov. No direction home: The true cost of routing around decoys. In *Network and Distributed System Security Symposium*, 2014.

[12] James Ball. NSA's PRISM surveillance program: how it works and what it can do , 2013. http://www.theguardian.com/world/2013/jun/08/nsa-prism-server-collection-facebook-google.

[13] R. Jansen and A. Johnson. Safely measuring tor. In *ACM Conference on Computer and Communications Security*, 2016.

[14] J. Karlin, D. Ellard, A. W. Jackson, C. E. Jones, G. Lauer, D. Mankins, and W. T. Strayer. Decoy routing: Toward unblockable internet communication. In *USENIX Workshop on Free and Open Communications on the Internet*, 2011.

[15] Z. Ling, J. Luo, W. Yu, M. Yang, and X. Fu. Extensive Analysis and Large-Scale Empirical Evaluation of Tor Bridge Discovery. In *IEEE INFOCOM*, 2012.

[16] K. Loesing. Sanitize TCP ports in bridge descriptors. https://trac.torproject.org/projects/tor/ticket/19317.

[17] K. Loesing and N. Mathewson. BridgeDB specification. https://gitweb.torproject.org/torspec.git/tree/bridgedb-spec.txt.

[18] J. Matherly. Shodan, 2013. https://www.shodan.io/.

[19] S. Matic, P. Kotzias, and J. Caballero. Caronte: Detecting Location Leaks for Deanonymizing Tor Hidden Services. In *ACM Conference on Computer and Communications Security*, 2015.

[20] J. McLachlan and N. Hopper. On the Risks of Serving Whenever you Surf: Vulnerabilities in Tor's Blocking Resistance Design. In *ACM Workshop on Privacy in the Electronic Society*, 2009.

[21] H. M. Moghaddam, B. Li, M. Derakhshani, and I. Goldberg. Skype-Morph: Protocol Obfuscation for Tor Bridges. In *ACM Conference on Computer and Communications Security*, 2012.

[22] M. Schuchard, J. Geddes, C. Thompson, and N. Hopper. Routing Around Decoys. In *ACM Conference on Computer and Communications Security*, 2012.

[23] K. Shubber. A simple guide to GCHQ's internet surveillance programme Tempora, 2013. http://www.wired.co.uk/news/archive/2013-06/24/gchq-tempora-101.

[24] R. Smits, D. Jain, S. Pidcock, I. Goldberg, and U. Hengartner. BridgeSPA: Improving Tor Bridges with Single Packet Authorization. In *ACM Workshop on Privacy in the Electronic Society*, 2011.

[25] Tor Project. CollecTor. https://collector.torproject.org/.

[26] Tor Project. Evaluate, possibly revise, and then implement ideas for TLS certificate normalization. https://trac.torproject.org/projects/tor/ticket/7145.

[27] Tor Project. Guide to run an obfs4 bridge. https://trac.torproject.org/projects/tor/wiki/doc/PluggableTransports/obfs4proxy.

[28] Tor Project. obfs3 (the threebfuscator). https://gitweb.torproject.org/pluggable-transports/obfsproxy.git/tree/doc/obfs3/obfs3-protocol-spec.txt.

[29] Tor Project. Obfsbridges should be able to "disable" their ORPort. https://trac.torproject.org/projects/tor/ticket/7349.

[30] Tor Project. Pluggable transports. https://www.torproject.org/docs/pluggable-transports.html.en.

[31] Tor Project. Tor Cloud: Tor bridges in the Amazon cloud. https://cloud.torproject.org/.

[32] Tor Project. Tor directory protocol, version 3. https://gitweb.torproject.org/torspec.git/tree/dir-spec.txt.

[33] Tor Project. Tor Manual. https://www.torproject.org/docs/tor-manual.html.en.

[34] Tor Project. On recent and upcoming developments in Pluggable Transports, 2014. https://blog.torproject.org/blog/recent-and-upcoming-developments-pluggable-transports.

[35] Tor Project, 2016. https://www.torproject.org/.

[36] Tor Project. Tor Metrics, 2016. https://metrics.torproject.org/.

[37] M. C. Tschantz, S. Afroz, D. Fifield, and V. Paxon. SoK: Towards Grounding Censorship Circumvention in Empiriscism. In *IEEE Symposium on Security & Privacy*, 2016.

[38] V. Tsyrklevich. Internet-wide scanning for bridges. https://lists.torproject.org/pipermail/tor-dev/2014-December/007957.html.

[39] L. Wang, K. P. Dyer, A. Akella, T. Ristenpart, and T. Shrimpton. Seeing through Network-Protocol Obfuscation. In *ACM Conference on Computer and Communications Security*, 2015.

[40] Z. Weinberg, J. Wang, V. Yegneswaran, L. Briesemeister, S. Cheung, F. Wang, and D. Boneh. StegoTorus: a Camouflage Proxy for the Tor Anonymity System. In *ACM Conference on Computer and Communications Security*, 2012.

[41] T. Wilde. Knock knock knockin' on bridges' doors, 2012. https://blog.torproject.org/blog/knock-knock-knockin-bridges-doors.

[42] P. Winter. Scramblesuit protocol specification, 2013. https://gitweb.torproject.org/user/phw/scramblesuit.git/tree/doc/scramblesuit-spec.txt.

[43] P. Winter, R. Ensafi, K. Loesing, and N. Feamster. Identifying and characterizing Sybils in the Tor network. In *USENIX Security Symposium*, 2016.

[44] P. Winter and S. Lindskog. How the Great Firewall of China is Blocking Tor. In *USENIX Free and Open Communications on the Internet*, 2012.

[45] P. Winter, T. Pulls, and J. Fuss. ScrambleSuit: a Polymorphic Network Protocol to Circumvent Censorship. In *ACM Workshop on Privacy in the Electronic Society*, 2013.