Survey on Bridge Discovery in Tor

Fucai Yu[1](✉), Ruoshui Zhou[1], Xuemeng Zhai[1], Youyang Qu[2], and Gaolei Fei[1]

[1] University of Electronic Science and Technology of China (UESTC), Chengdu, China
fcyu@Uestc.edu.cn
[2] Deakin University, Burwood, VIC 3125, Australia

Abstract. To prevent users from using Tor for anonymous communication, many regulatory agencies have blocked the IP addresses of public Tor routers in Tor networks, resulting in the interception of traffic to Tor public routers. Existing research solves this problem by introducing bridge nodes into the Tor network to avoid supervision: The bridge node is usually the entrance node of the Tor network, and its information is not completely public on the network, so it cannot be intercepted completely. This allows anonymous users to access the Tor network through the bridge node, which can effectively avoid Tor censorship. Nevertheless, many studies still focus on the discovery of Tor bridge nodes. The technology of bridge node discovery in Tor networks within recent years is summarized in this paper.

Keywords: Tor networks · Bridge · Tor relay nodes

1 Introduction

A Tor network [1], as shown in Fig. 1, is an overlay anonymous network in which each Tor router runs as a normal user-level process without any special privileges. Each user runs local software called an onion proxy to fetch directories, establish circuits across the network, and handle connections from user applications. These onion proxies accept TCP streams and multiplex them across the circuits. The Tor router on the other side of the circuit connects to the requested destinations and relays data. A Tor network relies on onion routing to guarantee anonymity. Onion routing [1], as shown in Fig. 2, is a distributed overlay network protocol designed to anonymize TCP-based applications like web browsing, secure shell, and instant messaging. Clients choose a path through the network and build a circuit in which each onion router in the path knows its predecessor and successor but no other nodes in the circuit.

Normal clients access the Tor core network directly through public Tor entry routers listed in the consensus file available at the official Tor website. To anonymously communicate with a web server, a normal client uses source routing and chooses a series of onion routers (generally three) from a downloaded consensus file. The selected onion routers construct an anonymous path along which a circuit will be set up incrementally by applying onion routing. Figure 1 depicts a circuit created incrementally along the path Source → Router A → Router B → Router C. Here, Routers A, B, and C also refer to entry, middle, and exit nodes, respectively.

W. Bao et al. (Eds.): ADHOCNETS 2021/TridentCom 2021, LNICST 428, pp. 317–326, 2022.
https://doi.org/10.1007/978-3-030-98005-4_23

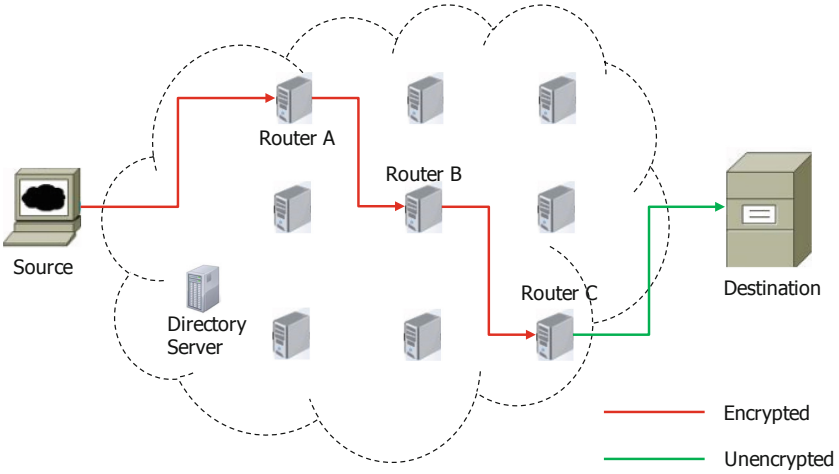

Fig. 1. Sample Tor circuit between source and destination.

Tor has been commonly used for resisting various forms of censorship [2]. However, Tor uses source routing for communication privacy, and the information of all Tor routers is available to clients and publicly listed on the Internet [3]; thus, blocking Tor is as simple as blocking connections to those known Tor routers. To resist the censorship blocking of public Tor routers, bridges were introduced in Tor. A bridge can act as the first hop relaying user traffic into the core Tor network, i.e., entry node (Router A) in Fig. 1. The bridge information is not listed on the Internet. As described in [4], the bridge population significantly varies over time: It has steadily grown from 2.8K active public bridges in July 2012 to a maximum of 12.7K in July 2014; it began declining in January 2015, falling to 5.3K by April 2016. A few bridge pools exist and some are stored on the bridge https and email servers. A user can access the bridge https server or send a Google/Yahoo email to the bridge email server to retrieve three bridges at one time. Bridges are also distributed through various social networks.

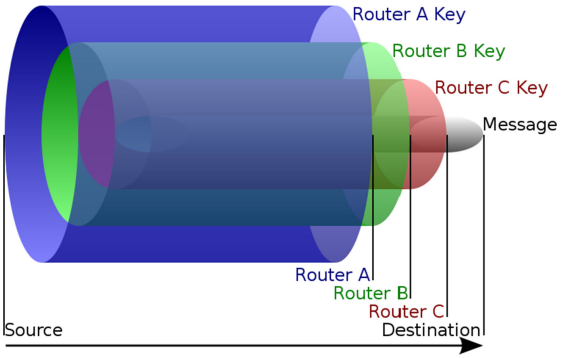

Fig. 2. Tor onion routing, Routers A, B, and C correspond to those in Fig. 1, respectively.

For subjects such as blocking Tor, Tor censorship, finding available bridges, Tor network analysis, performance improvement, and bridge concealment, a number of studies have focused on bridge discovery. Although Arma presented 10 possible ways to obtain bridges on Tor's official website [5], he did not describe the exact bridge acquisition method in detail. In this paper, a brief survey of bridge discovery with respect to bridge categories and discovery schemes is conducted.

2 Background

The components of Tor and bridges are reviewed in this section. As described in [6], a Tor client runs an onion proxy (OP) to anonymize the client data into Tor. Tor uses a multi-hop proxy mechanism to protect user communication privacy: The client first uses a weighted random routing algorithm to select three relay nodes from the consensus file as the Tor network entry, middle, and exit nodes and then establishes a circuit with these relay nodes hop by hop. As shown in Fig. 2, the client encapsulates the sent data with three-layer encryptions. In the process of data transmission, the entry, middle, and exit nodes sequentially decrypt and restore the plaintext and then send it to the destination. The client's user privacy is guaranteed by using onion routing, under which any relay node cannot know the IP address of both the client and the destination server, and the client IP address is also unknown to the destination server. The public relay node list is stored in the accessible consensus file; therefore, to block Tor network communication, one only needs to block the IP address of the public relay node IP obtained from the accessible consensus file. Bridge nodes were introduced into the Tor project to resist such supervision. The Tor project provides all the source code of the Tor system. Users can create bridge nodes by downloading Tor source code and use bridge nodes instead of public relay nodes as entry nodes. Since the bridge node information is not accessible, under this situation, regulatory agencies cannot block Tor communication effectively. Bridge nodes are classified into two categories: public bridge nodes and private bridge nodes [4]. Public bridge node information is stored in the Tor network directory server, which only provides three public bridge nodes to a user at a time. Therefore, it is not easy to obtain the information of all bridge nodes. Of course, users can obtain a large amount of bridge node information through distributed enumeration. The private bridge node information is only kept by the creator and cannot be obtained by other users.

Although Tor can use bridge nodes to circumvent the supervision based on IP blocking, supervisors can still identify Tor traffic through deep packet inspection (DPI) technology [7]. To make Tor communication characters fuzzy in using bridge node environments, a transmission plug-in called an obfs, or obfuscation, proxy was developed by the Tor project. The latest version of obfs4 [8] uses BridgeDB to implement key exchange based on bridge authentication. The client queries the bridge node through BridgeDB and obtains its IP address, node ID, and public key information. Only by matching these three conditions at the same time can it pass the identity verification of the obfs4 node and establish a connection. obfs4 not only effectively confuses Tor traffic but also realizes, with bridge authentication, that it can resist active detection attacks and man-in-the-middle attacks.

3 Bridge Discovery Schemes

The discovery methods of bridge nodes are classified and summarized in this section. The specific methods are as follows.

3.1 Normal Bridge Enumeration Mechanism

Ling *et al*. elaborated on and verified the method of obtaining bridge node information in batches through email and http requests in [6]. An attacker can use a Yahoo or Gmail account to send an email to the bridge email server (bridges@torproject.org) with the line "get bridges" in the body of the mail. The bridge email server promptly replies with three distinct bridges. To avoid malicious enumeration, the bridge email server only replies with one email to an email account each day. Alternatively, the user can access the bridge website (https://lbridges.torproject.org) to obtain three bridges. To avoid malicious enumeration, the https server distributes three bridges to each 24-bit IP prefix each day as well.

The authors verified the effectiveness of the above method of obtaining bridge nodes through emails and http requests. Since each mailbox can only obtain three bits of bridge node information per day, obtaining more bridge node information requires a large number of mailboxes to send bridge node request emails, but an IP address can only apply for one Yahoo mailbox. To solve this problem, the authors applied for Yahoo mailboxes through more than 500 PlanetLab nodes and also used more than 500 exit nodes of the Tor network as agents to apply for Yahoo mailboxes. In the end, the authors applied for more than 2,000 Yahoo mailboxes in total and enumerated more than 1,500 bridge nodes within 22 d through these mailboxes. In addition, the authors also used these PlanetLab nodes and Tor exit nodes to retrieve the bridges from the bridge website (https://lbridges.torproject.org) and obtained more than 550 bridge nodes within 37 d. McLachlan described a similar method in [9]. The advantage of the above bridge node enumeration method is that the fingerprint information of the bridge node can be obtained. With the fingerprint information, these bridge nodes can be used to construct anonymous communication links for anonymous communication. To avoid the information of all bridge node being enumerated, the bridge website may only provide part of the bridge node information for users to obtain within a period of time. In addition, only public bridge nodes can be obtained in this way; private bridge nodes cannot be obtained using this method.

3.2 Bridge Inference by Malicious Tor Middle Routers

Ling *et al*. also described a method of using malicious middle nodes to infer bridge nodes in [6]. The anonymous circuit of the Tor network is composed of entry, middle, and exit nodes. The entry node may be a general public Tor router or a bridge node. The anonymous traffic received by the middle node comes from the entry node. A malicious middle node can obtain the IP addresses of all entry nodes by extracting the source IP of the received data packets. By excluding the IP addresses of the known public routers, an attacker obtains the IP information of the bridge nodes. This method requires that the middle node be a controlled node, and the identified bridge nodes only include those

transmitting anonymous communication traffic through the controlled middle node. To identify more bridge nodes, the controlled middle node must have a higher bandwidth to attract more anonymous communication traffic.

3.3 Tor Bridge Discovery Through Internet-Wide Scans

Ports 443 and 9001 are common ports for Tor bridges and relays, and the TLS protocol is used for encryption and authentication between bridges/relays. Based on the above characteristics, Durumeric *et al*. [10] and Tsyrklevich [11] successively proposed a bridge node discovery method based on Internet-wide scans. Durumeric *et al*. exploited the ZMap tool to perform Internet-wide scans on ports 443 and 9001 and applied a set of heuristics to identify likely Tor nodes. For hosts with one of these ports open, they performed a TLS handshake using a specific set of cipher suites supported by Tor's "v1 handshake." When a Tor relay receives this set of cipher suites, it will respond with a two-certificate chain. The signing ("Certificate Authority") certificate is self-signed with the relay's identity public key and uses a subject name of the form "CN = www.X.com," where X is a randomized alpha-numeric string. This pattern matched 67,342 hosts on port 443 and 2,952 hosts on port 9001. Durumeric *et al*. then calculated each host's identity fingerprint and checked whether the SHA1 hash appeared in the public Tor metrics list for bridge pool assignments. Hosts that were found matched 1,170 unique bridge fingerprints on port 443 and 419 unique fingerprints on port 9001, with a combined total of 1,534 unique fingerprints (some were found on both ports). From the bridge pool assignment data, they found that 1,767–1,936 unique fingerprints were allocated at any given time in the recent past, which suggests that they were able to identify 79%–86% of allocated bridges at the time of the scan. The unmatched fingerprints in the Tor metrics list may correspond to bridges missed, offline bridges, or bridges configured to use a port other than 9001 or 443. Based on the work of Durumeric *et al*., Tsyrklevich [11] identified pluggable transport (PT) [4]-enabled bridges. A PT is just a wrapper for the Tor protocol that transforms the Tor traffic flowing between clients and bridges to prevent DPI attacks. Experimental results have also proven the effectiveness of the Internet-wide scan method in bridge recognition. Wilde also described a similar method in [12]. The advantage of the above method is that there is no need to deploy malicious Tor relay nodes. It only needs to detect the 443 and 9001 ports of a large number of IP addresses and further verify the detection results. The disadvantage of this method is its poor timeliness. Because the set of active bridges is constantly changing, the data would be stale by the time a long-running scan was complete.

3.4 Tor Bridge Identification Through DPI

In 2011, Iran added a filter rule to its border routers that recognized Tor traffic and blocked it. Arma investigated the situation in [13] and found the following. (1) Tor tries to make its traffic look like a web browser talking to a https web server, and the characteristic of Tor's SSL handshake was the expiry time for SSL session certificates. (2) Tor's SSL handshake rotates the session certificates every 2 h, whereas normal SSL certificates received from a certificate authority typically last a year or more. The fix was to simply write a larger expiration time on the certificates, so the current certificates in

use have more plausible expiry times. Another DPI-based method developed by Winter *et al*. [14] and Wilde [12] is to check a special cipher list in the Tor TLS handshake. The cipher list is part of the TLS client hello, which is sent by the Tor user to the relay or bridge after a TCP connection has been established. This particular cipher list appears to be unique to Tor: c0 0a c0 14 00 39 00 38 c0 0f c0 05 00 35 c0 07 c0 09 c0 11 c0 13 00 33 00 32 c0 0c c0 0e c0 02 c0 04 00 04 00 05 00 2f c0 08 c0 12 00 16 00 13 c0 0d c0 03 fe ff 00 0a 00 ff. Through these methods, a Tor TLS handshake can be detected, thereby determining whether the entry node is either a public relay or a bridge.

3.5 Tor Bridge Detection Based on TCP SYN Connections

Yang *et al*. [15] proposed a bridge detection method based on the following observation: When connecting to a Tor network via bridges, the client first creates a series of non-blocking sockets and then connects those sockets one after another in a short period for establishing TCP connections to chosen bridges. The chosen bridges include two parts: bridges configured by users and bridges cached by Tor software. As with most applications, the Tor client does not bind pre-determined source ports for those connections. Conversely, it is the operating system that assigns ephemeral source ports for connections to chosen bridges. Such ephemeral source ports are usually consecutively allocated in widely used operating systems like Windows platforms. As a result, multiple SYN packets with consecutive source ports will be sent almost simultaneously, destined for different bridges. Therefore, if multiple SYN packets from the same IP address with consecutive source ports are observed, then the destination IPs are extracted, and, if at least one destination IP belongs to a known bridge set, then all the other destination IPs are inferred to be bridges. This is called the opportunism bridge detection method and requires a large known bridge set.

3.6 Tor Bridge Detection Based on Flow Classification or Identification

The idea of the bridge identification method based on Tor traffic classification and identification is to identify Tor anonymous communication traffic, extract the source and destination IP addresses of the traffic at first, and then exclude the IP addresses of the known public routers from the accessible consensus file to obtain the bridge node information. Several studies have focused on Tor flow classification [16, 17]. In [16], Shahbar *et al*. aimed to analyze the amount of information that can be extracted from the encrypted Tor traffic without decrypting the traffic. They employed two different approaches for the classification of user activities. The first approach is flow level classification, and it depends on analyzing the TCP communication between the user and the Tor relay to predict the type of user activities in the encrypted traffic. The second approach is circuit-level classification. The encrypted circuits have characteristics that can be extracted and calculated to classify the type of traffic in the circuits.

In [18], traffic analysis was used to discover the identity of the user using the Tor network. The analysis depends on the size of the packet transmitted on the network from the web server through the router to the user. Tor has a fixed cell size, but the packet size can vary. The authors of [18] used padding with one bit to mark the packets so that they could be traced back at the receiver side. They reported that 10 packets are enough to

get reasonable detection with a low number of false positives. The length of the padding will force the Tor router to use the known number of cells. The Tor cell size is 512 bytes, which means if the data size is more than 512 bytes, then it must be fragmented to fit into the cell size. This enabled the authors to mark the client receiving these cells as the client having access to the server.

To avoid censorship, obfs4 has been widely deployed to obscure the flow between a Tor client and bridge. In [19], He *et al*. proposed a scheme for obfs4 traffic detection based on two-level filtering. They sequentially utilized coarse-grained fast filtering and fine-grained accurate identification to achieve high-precision, real-time recognition of obfs4 traffic. In the coarse-grained filtering phase, they used a randomness detection algorithm to detect the randomness of the handshake packet payload in the communication and used the timing-sequence characteristics of the packet in the handshake process to remove other interference traffic. In the fine-grained identification phase, they analyzed its statistical feature on a large amount of obfs4 traffic and used classification algorithms to identify the obfs4 traffic. Their experimental results show that the accuracy for identifying obfs4 traffic is above 99% when using a support-vector-machine algorithm, which indicates that obfs4 cannot effectively counteract traffic analysis attacks in practical applications. Once an instance of obfs4 traffic was identified, the corresponding IP address could be further confirmed as whether it was a bridge based on the port.

Through the above methods, one can first identify which users are using a Tor network for anonymous communication, then extract the destination IP addresses of anonymous traffic sent by these users, and finally remove public relays from them to obtain the bridge node information.

4 Discussion

Among the above studies on bridge recognition, only the bridge node information obtained based on http and e-mail requests described in [6] contains the fingerprint of the bridge nodes. According to the obtained bridge node fingerprint information, the obtained bridge node can be used to construct a circuit for anonymous communication. In addition, the connection configuration window of the Tor browser also provides the function of obtaining three bridge nodes and their fingerprints. Although other bridge nodes cannot obtain the fingerprint information of a bridge node, the obtained IP address and port information of the bridge nodes can meet the requirements of blocking Tor, Tor censorship, finding available bridges, Tor network analysis, performance improvement, bridge concealment, etc. The above-mentioned bridge node discovery algorithms are summarized in Table 1. In the table, “bridge availability” indicates that the method can obtain fingerprint information of bridge nodes and that the fingerprint information can be used to construct an anonymous communication circuit; “bridge comprehensive” indicates the proportion of bridge nodes that can be discovered by the particular method in the total bridge nodes of the Tor network; “discovery efficiency” indicates the efficiency of the method, i.e., the number of bridge nodes discovered per unit time; “method effectiveness” indicates whether the bridge node discovery methods are still available; and “method precision” indicates the proportion of real bridge nodes among the bridge nodes discovered by the listed methods.

Table 1. Comparison of bridge discovery methods.

Study		Bridge availability	Bridge comprehensive	Discovery efficiency	Method effectiveness	Method precision
Normal mechanism	Ling *et al.* [6]	Yes	Middle	High	Yes	High
	McLachlan *et al.* [9]	Yes	Middle	High	Yes	High
By Tor middle router	Ling *et al.* [6]	No	Low	Low	Yes	High
Internet-wide scans	Durumeric *et al.* [10]	No	High	Low	Yes	High
	Tsyrklevich *et al.* [11]	No	High	Low	Yes	High
Through DPI	Arma [13]	No	Low	Low	No	High
	Wilde [12]	No	Low	Low	No	High
	Winter *et al.* [14]	No	Low	Low	No	High
TCP SYN based	Yang *et al.* [15]	No	Low	Low	Yes	Low
Flow classification or identification	Shahbar *et al.* [16]	No	Low	Low	Yes	Middle
	Shahbar *et al.* [17]	No	Low	Low	Yes	Middle
	Ling *et al.* [18]	No	Low	Low	Yes	Middle
	He *et al.* [19]	No	Low	Low	Yes	Middle

Through the above comparison, one can find that the bridge node discovery methods mainly include the following: 1) The actual and available bridge node information can be obtained by using http and email requests; 2) the method of middle malicious forwarding nodes requires the attacker to deploy controllable middle forwarding nodes and can only identify the bridge nodes based on the monitored traffic; 3) although more comprehensive bridge node information can be obtained through Internet-wide scans, this takes a long time and requires further verification of suspected bridge nodes, so the real-time performance is poor; 4) the extraction of bridge node information through DPI depends on the particular cipher list used in Tor TLS negotiation; 5) the TCP SYN–based bridge node discovery method relies on the allocation of ephemeral source ports for connections to the operating system-chosen bridges, and if the operating system modifies the source port allocation strategy, this method will then not achieve the expected effect; and 6) the method based on flow classification or identification mainly realizes bridge node identification based on anonymous communication traffic classification, and its accuracy depends on the accuracy of the flow classification or identification method.

5 Conclusions

This article focuses on the discovery of bridge nodes in the Tor network, and several bridge node discovery methods are analyzed and summarized. Analysis results show that although the Tor project tries to conceal anonymous users through bridges, the concealment of bridge nodes is not perfect, and the existence of bridge nodes can still be discovered through a variety of methods. Future research should further enhance the concealment of bridges, with the following goals: 1) Improve the distribution mechanism of bridge nodes, increase the verification mechanism, and reduce the risk of public bridge

information being obtained through a large number of enumerations through http and email servers; 2) encourage users to establish and use bridges, so that the greater the number of bridge nodes, the stronger the concealment; 3) in addition to Meek [20] and obfs technology, new Tor traffic camouflage methods should be studied to reduce the risk of Tor traffic being identified; and 4) many web services prohibit Tor access by blocking the IP of the Tor exit node, the information of which is public. Therefore, the use of exit bridges [21, 22] to bypass server-side censorship—which has rarely been researched—should be studied.

References

1. Dingledine, R., Mathewson, N., Syverson, P.: Tor: the second-generation onion router. In 13th USENIX Security Symposium, San Diego, pp.1–17 (2004)
2. Phobos: Tor and Censorship: lessons learned. https://blog.torproject.org/tor-and-censorship-lessons-learned
3. Tor Node List. https://www.dan.me.uk/tornodes
4. Matic, S., Troncoso, C., Caballero, J.: Dissecting tor bridges: a security evaluation of their private and public infrastructures. In: Conference of Network and Distributed System Security Symposium 2017, San Diego, pp. 1–15 (2017)
5. Arma: Research problems: Ten ways to discover Tor bridges. https://blog.torproject.org/research-problems-ten-ways-discover-tor-bridges
6. Ling, Z., Luo, J., Yu, W., et al.: Tor bridge discovery: extensive analysis and large-scale empirical evaluation. IEEE Trans. Parallel Distrib. Syst. **26**(7), 1887–1899 (2015)
7. He, G., Yang, M., Luo, J., Gu, X.: A novel application classification attack against Tor. Concurr. Comput. Pract. Exp. **27**(18), 5640–5661 (2016)
8. Angel, Y.: obfs4 (The obfourscator). https://github.com/Yawning/obfs4/blob/master/doc/obfs4-spec.txt
9. McLachlan, J., Hopper, N.: On the risks of serving whenever you surf: vulnerabilities in Tor's blocking resistance design. In: Proceedings of the 8th ACM workshop on Privacy in the electronic society, Chicago, pp. 31–40 (2019)
10. Durumeric, A., Wustrow, E., Halderman, A.: ZMap: fast Internet-wide scanning and its security applications. In: the Proceedings of the 22nd USENIX Security Symposium. 14–16 August 2013, Washington, D.C. pp. 605–619 (2013)
11. Tsyrklevich, V.: Internet-wide1976 scanning for bridges. https://lists.torproject.org/pipermail/tor-dev/2014-December/007957.html
12. Wilde, T.: Great Firewall Tor Probing. https://gist.github.com/da3c7a9af01d74cd7de7
13. Arma.: Iran blocks Tor. https://blog.torproject.org/iran-blocks-tor-tor-releases-same-day-fix
14. Winter, P., Lindskog, S.: How China Is Blocking Tor. 2012. https://arxiv.org/abs/1204.0447v1
15. Yang, M., Luo, J., Zhang, L., Wang, X., Fu, X.: How to block Tor's hidden bridges: detecting methods and countermeasures. J. Supercomput. **66**(3), 1285–1305 (2012). https://doi.org/10.1007/s11227-012-0788-4
16. Shahbar, K., Zincir-Heywood, A.N.: Benchmarking two techniques for tor classification: flow level and circuit level classification. In: 2014 IEEE Symposium on Computational Intelligence in Cyber Security (CICS), Orlando, pp. 1–8 (2014)
17. Shahbar, K., Zincir-Heywood, A.N.: Traffic flow analysis of tor pluggable transport. In: 2015 11th International Conference on Network and Service Management (CNSM), Barcelona, Spain, pp.178–181 (2015)

18. Ling, Z., Luo, J., Yu, W., Fu, X.: Equal-sized cells mean equal-sized packets in Tor? In: Proceedings of IEEE International Conference on Communications ICC 2011, Kyoto, Japan, pp.1–6 (2011)
19. He, Y., Hu, L., Gao, R.: Detection of tor traffic hiding under obfs4 protocol based on two-level filtering. In: 2019 2nd International Conference on Data Intelligence and Security, South Padre Island, pp.195–200 (2019)
20. Fifield, D., Lan, C., Hynes, R., et al.: Blocking-resistant communication through domain fronting. Proc. Privacy Enhan. Technol. **2015**(2), 46–64 (2015)
21. Zhang. Z., Subramanian, K., Zhou, W., Sherr, M.,: Ephemeral exit bridges for tor. In: 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2020, Spain, pp. 253–265 (2020)
22. Zhang, Z., Zhou, W., Sherr, M.: Bypassing tor exit blocking with exit bridge onion services. In: The ACM Conference on Computer and Communications Security (CCS'20), Virtual Event, 2020, pp. 3–16 (2020)