B-Tor:Anonymous Communication System Based on Consortium Blockchain

Dawei Xu
Beijing Institute of Technology
Jiaqi Gao
Changchun University
Liehuang Zhu (liehuangz@bit.edu.cn)
Beijing Institute of Technology
Feng Gao
Beijing Institute of Technology
Yang Han
Beijing Institute of Technology
Jian Zhao
Changchun University

Research Article

Keywords: Tor, decentralization, Hyperledger Fabric, traceability

Posted Date: June 1st, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1680101/v1

B-Tor:Anonymous Communication System Based on Consortium Blockchain

Dawei Xu[1,2], Jiaqi Gao[2], Liehuang Zhu[1*], Feng Gao[1], Yang Han[1] and Jian Zhao[2]

[1*]School of Cyberspace Security, Beijing Institute of Technology, Beijing, 100081, China.
[2]College of Cyber Security, Changchun University, Changchun, 130022, China.

*Corresponding author(s). E-mail(s): liehuangz@bit.edu.cn;
Contributing authors: xudw@ccu.edu.cn; judgegao06@gmail.com; gaofengbit@foxmail.com; y_han@outlook.com; zhaojian@ccu.edu.cn;

Abstract

Tor, an anonymous communication system that protects the privacy of individuals, is gaining more and more attention. However, there are some problems with the Tor system, such as: security issues such as vulnerability to de-anonymisation attacks, centralised and poorly scalable directory servers, bot nodes disrupting the system; social issues such as illegal transactions in the dark web disrupting social security; performance issues such as high communication latency and low system throughput. To address these issues, a new anonymous communication system, B-Tor, is designed and implemented based on the Tor architecture model and the mainstream consortium blockchain architecture, Fabric, which uses the decentralized feature of blockchain to solve the security problems in Tor. It also provides a solution to the problem of unregulated and illegal transactions in the system. The system meets the needs of users for anonymous communication while reducing the harm caused by the misuse of anonymous communication technology. The experiments show that B-Tor can achieve a decentralised structure by reading smart contracts to obtain consensus files, and that it can successfully receive consensus files 790629 times in

600s of testing time, with a success rate of 98%, an average throughput of 1312.9TPS and an average latency of 0.1s. The experimental results show that B-Tor has high efficiency, high load and low latency.

Keywords: Tor, decentralization, Hyperledger Fabric, traceability

1 Introduction

With the rapid development of the Internet, Internet users take personal data privacy more seriously. However, Internet users still face many security risks. For example, sent messages are stolen, stored data is leaked, communication identities are identified, user locations are tracked, etc[1]. Due to many of the above problems and the rise of virtual currencies in recent years, anonymous communication systems have grown from small-scale use to mass-market software with millions of daily usage. Anonymous communication systems such as Tor, I2P, Freenet are widely used[2, 3].

As anonymous communication systems gradually enter the public eye, more and more researchers are interested in anonymous communication systems. Its research work is mainly in the aspects of deployment cost, congestion control, scalability, anti-eavesdropping, anti-blockade, anti-tracking, and robustness[4].

But the anonymous communication system is also a double-edged sword, and the misuse of anonymous communication technology will bring instability to the whole society. Under the protection of anonymous communication technology, criminals carry out various criminal activities. The most notorious of them is the Dark Web, which is full of illegal transactions such as extortion, drugs, guns, and human trafficking. At present, many scholars have carried out extensive research on transaction traceability and de-anonymization on the dark web[5]. This unregulated anonymous communication system brings more insecurity factors to society, so it is subject to various countries' scrutiny, boycott and blockade[6–9].

Tor (the second generation of onion routing)[10], the anonymous communication system with the largest number of users. It is filled with a large number of illegal transactions, and many studies have shown that Tor has security problems (explained in Section 2). Currently, various countries mainly adopt the blocking methods for the existing Tor anonymity system: filtering Tor traffic through traffic identification, blocking the IP addresses of Tor routing nodes, etc[11]. In this way, although the malicious behavior of illegal users is restricted and social stability is increased, some legitimate users who need anonymity cannot use it.

In view of the above problems, it is necessary to satisfy an anonymous communication system that is licensed by the government and state departments, traceable to the source and meets the anonymity needs of legitimate users. This paper combines the characteristics of blockchain, designs and implements an anonymous communication system B-Tor for tracking and tracing illegal behaviors. The emergence of the traceable anonymous communication system can solve the pain points of cybercrime brought by the current public anonymous system, and at the same time, it can also ensure the privacy and data security of the daily communication of legitimate users.

Contributions of this article:

This paper takes the multi-hop proxy mechanism of the anonymous communication system Tor as the basis of anonymous communication and combines the IBM company's open-source consortium blockchain platform Hyperledger Fabric as the underlying framework to design an anonymous system B-Tor based on consortium blockchain. (1)Innovatively proposed a traceable anonymous communication system model. (2) Solved the centralization problem of Tor system directory server. (3) Solved the problem of Tor consensus file leakage. (4) Solved the untrustworthy problem of Tor intermediate nodes.

The Section 2 introduces the anonymous communication system Tor, its working principle, and blockchain technology. And analyzed the vulnerability of Tor, and finally verified its vulnerability through the proposal given by the official Tor.

The Section 3 introduces how B-Tor, an anonymous communication system based on consortium blockchain, responds to the threat model and its design concept.

The Section 4 introduces the system architecture model of B-Tor and introduces the specific details of the implementation of each module in the system. These include: (1) how the system implements user registration and initialization; (2) how the client builds links for communication (3) how intermediate nodes join the network and upload their information to update the network topology.

The Section 5 is optimized for the B-Tor architecture. Describes how to anonymize registration and how to de-anonymize when targeting abuse of the anonymity network. At the same time, node management and incentive mechanism are introduced to ensure the stable and safe operation of the system.

In the Section 6, the experimental verification of the above system model is carried out. By writing chaincodes, the generation, reading, and updating of consensus files in the anonymous communication system B-Tor are realized. And test the communication efficiency and network load of the anonymous communication system in the Fabric framework.

An anonymous communication system is a communication system built on the Internet that uses data forwarding, content encryption, traffic obfuscation, and other technologies to hide the sender's identity. Since the anonymous communication system can hide the service mechanism and anonymize the deployment. Therefore, it has the characteristics of difficult node discovery, difficult user monitoring, and difficult communication relationship confirmation. As a result, illegal and criminal activities based on the anonymous communication system emerge endlessly. For example, "Dark Web", "Silk Road", AlphaBay and other online black markets. At the same time, it has also become a shelter for illegal transactions such as WannaCry ransom transactions and BitCoin money laundering transactions. At present, there is no complete and reasonable method for anonymous communication and darknet governance.

Tor (the second-generation onion route) is a widely used anonymous communication system. Its core "onion routing" was proposed by the US Naval Research Laboratory in the 1990s. The Tor system is based on a multi-hop proxy mechanism to protect the anonymity of users[12].

2 Background and related work

2.1 Anonymous communication system Tor

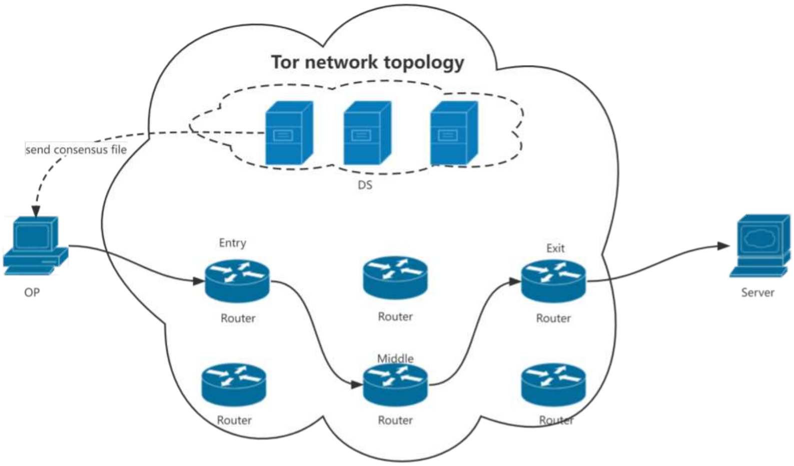


Fig. 1: Tor network architecture model

The Tor system is mainly composed of a large number of distributed relay nodes. As shown in Figure 1, the overall system is mainly divided into the following five parts[13]: 1) Client (OP): a local program running on the user's operating system, providing anonymous proxy services for users. 2) Directory Server (DS). There are 10 worldwide. The function is to collect and organize node information in the network, generate consensus files according to internal algorithms, actively detect the reachability of nodes, update consensus files according to reachability, and allow clients to access and obtain consensus files, to complete node selection. and link establishment. 3) The Onion Router (OR). Data relay nodes in the Tor anonymous network. Tor's default anonymous link consists of three ORs, namely the entry node (Entry), the intermediate node (Middle), and the exit node (Exit). 4) Hide the server. Provides TCP application services such as Web and IRC. The hidden server is protected by Tor anonymity, and a Tor client must be used to be able to access its TCP application services. 5) Hide service directory server. The hidden service directory server stores and provides the client with node information such as the introduction point (IPO) and public key of the hidden server.

2.2 Blockchain technology

Most of the network systems in the traditional mode adopt the B/S architecture or the C/S architecture. The common feature of these architectures is that a centralized server is required. Each user needs to interact with the central server, including uploading and downloading data. However, in the process of data storage and transmission, the centralized management system has security and trust issues, and the vulnerability of the central server will also affect the use of users across the network. If the central server is paralyzed, it will affect the use of the entire network. Blockchain technology originated from a peer-to-peer cash transaction system proposed by Satoshi Nakamoto in 2008[14]. Blockchain technology integrates various security mechanisms such as P2P network, cryptographic algorithm, consensus mechanism, link structure, etc., and realizes a decentralized system that ensures the credibility of the entire network.

A blockchain can be defined as an immutable ledger that records transactions and maintains a mutually distrusting distributed network. Each peer has a copy of the ledger. Peers perform a consensus protocol to validate transactions, group them into blocks, and build hash chains over the blocks. This process forms a ledger by ordering transactions, which is required for consistency. Blockchain has appeared in many applications such as Bitcoin, Ethereum, Hyperledger Fabric, etc., and is widely regarded as a promising technology for running trusted transactions in the digital world[15].

2.3 Tor vulnerability analysis

2.3.1 Attacks on Tor

At present, there are many kinds of attacks on Tor. Cambiaso E et al. classified various attack methods against Tor in 2019, attacking the client, attacking the server, and attacking the overall network[16]. Many researchers have shown that the Tor system has the following vulnerabilities.

DoS attack: Tor community developer Rob Jansen et al. proposed a sniper attack in 2014, an extremely low-cost and extremely destructive denial-of-server attack[17]. At the same time, in 2019, it was explained that Tor faced DoS attacks and experimentally quantified the cost of each attack and its impact on Tor performance[18]. Mane Y D et al proposed an efficient technique for detecting Tor server DDoS attacks in 2020[19]. In 2021, Rui Wang et al. experimentally demonstrated the effectiveness of DoS attacks and discussed the defense strategies for this attack[20].

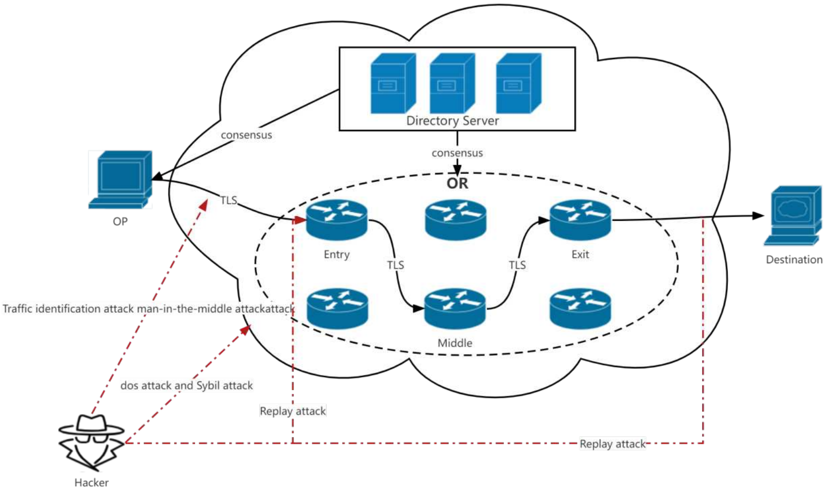


Fig. 2: Tor attack location

Traffic identification attack: By capturing encrypted data traffic, preprocessing the data traffic, extracting traffic features from it, and finally using machine learning algorithms to select features and make classification predictions. Basyoni L conducted a traffic analysis attack on Tor from an attacker's perspective in 2020 and stated that this attack applies to the vast majority of Tor scenarios[21]. Sun Xueliang expounds on tag-oriented and multi-tag website traffic identification attacks and discusses their principles in 2021[22]. Lashkari A H et al. proposed a temporal feature-based Tor fingerprinting attack in 2017 and released a test dataset[23]. In 2020,Florian Platzer proposes a Tor traffic analysis method that allows attackers to de-anonymize any hidden service in less than 12.5 days, which poses a threat to online anonymity[24].

Man-in-the-middle attack: For the Tor system, by deploying a "malicious man-in-the-middle", a man-in-the-middle attack is launched on the link in the anonymous communication system. The simplest attack is to disguise a node as a man-in-the-middle between two nodes in the link, thereby destroying the communication link. Sanatinia A et al. in 2017 demonstrated that an attacker who cracked the private key can launch an attack on a hidden directory server[25].

Sybil attack: The witch attacker forges enough relay nodes, and the user has a high probability that the 3 relay nodes selected in a row are all nodes forged by the witch attacker, which will cause all the user's data to be decrypted. Philipp Winter et al. developed a sybilhunter Sybil attack tool in a real network in 2016 and experimentally proved that Tor cannot resist Sybil attack[26].

Replay attack: In 2008, Pries et al. proposed an anonymous network source tracing technology based on replay attack, which adopts the scenario mode of AES calculator encryption mode AES-CTR. The attacker intentionally modifies the value in the counter, resulting in an asynchronous situation, which in turn exposes the communication status of the network. By controlling the nodes in the anonymous network, replaying the communication data causes the node to fail to receive the data so that the communication relationship between the two communicating parties is also manifested[27].

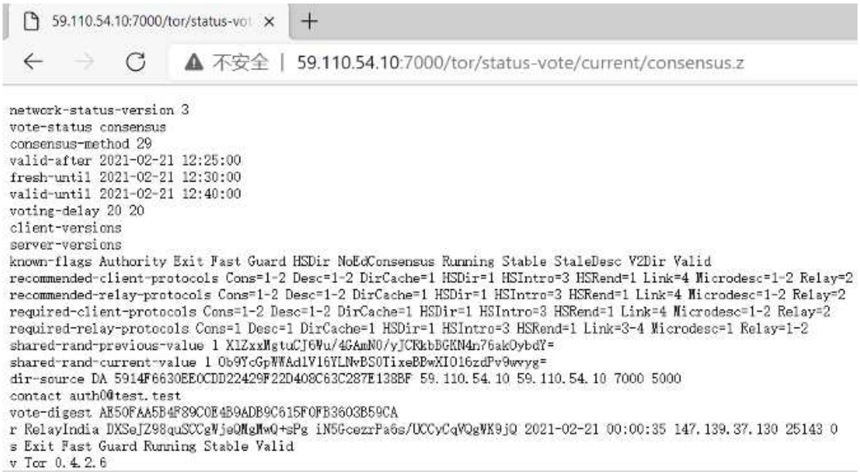
Attacks against Tor are not limited to the above-mentioned attacks. Figure 2 depicts the Tor system, where attackers launch attacks, and the main attacks they face. Attackers can launch traffic identification attacks and man-in-the-middle attacks by detecting the traffic from the client to the entry node. Replay attacks by intercepting traffic at ingress and egress. DoS attacks against single or multiple servers in the Tor network. By controlling multiple Tor relay nodes, launching Sybil attacks, etc.

2.3.2 Tor Architecture Vulnerability

Disclosure of consensus file : The IP address of the directory server in the Tor system is exposed to the whole network. Anyone who wants to obtain the consensus file can obtain the consensus file of the entire system by sending a GET request to the directory server. Therefore, after obtaining the IP of the directory node, the attacker can simulate the client to send a Get request to the directory server. You can directly enter this type of URL http://IP:port/tor/status-vote/current/consensus.z in the browser to get the plaintext consensus file. Part of the consensus file is shown in Figure 3. The plaintext consensus file will contain the following sensitive information: IP address, region, bandwidth, Tor version, etc. of the intermediate nodes of the system. Once the attacker obtains the information of all relay nodes in the whole network, he can launch DOS, cryptography, and other attacks on it in a targeted manner. This behavior of exposing distributed nodes to the entire network has obvious security risks.

Directory server centralization: Tor directory servers are 10 authoritative directory servers officially formulated by Tor. Tor authoritative directory servers are distributed in 6 countries in North America and Europe, 5 in North America, and 5 in Europe, including 4 in the United States, 1 in Canada, 1 in the Netherlands, 1 in Austria, 2 in Germany, and 1 in Sweden. The specific bandwidth and update time and other information are shown in Figure 4 [28].

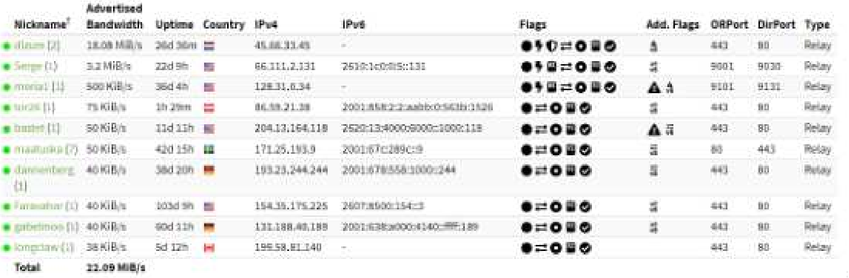
It can be seen that 7 of the 10 directory servers support IPv6 address access, and all the information of the nodes is public, such as the directory server's address, running time, bandwidth resources, and so on. The directory server has functions such as measuring node information and voting to generate consensus files. Attackers can attack these public directory servers. If they control more than half of the directory servers, they can tamper with the consensus files and destroy the anonymity of the overall system. At the same time, regulators can block the IP addresses of the above directory servers. Using this blocking method will make users unable to access anonymous communication networks.



59.110.54.10:7000/tor/status-vo ×
+
不安全 | 59.110.54.10:7000/tor/status-vote/current/consensus.z

```
network-status-version 3
vote-status consensus
consensus-method 29
valid-after 2021-02-21 12:25:00
fresh-until 2021-02-21 12:30:00
valid-until 2021-02-21 12:40:00
voting-delay 20 20
client-versions
server-versions
known-flags Authority Exit Fast Guard HSDir NoEdConsensus Running Stable StaleDesc V2Dir Valid
recommended-client-protocols Cons=1-2 Desc=1-2 DirCache=1 HSDir=1 HSIntro=3 HSRend=1 Link=4 Microdesc=1-2 Relay=2
recommended-relay-protocols Cons=1-2 Desc=1-2 DirCache=1 HSDir=1 HSIntro=3 HSRend=1 Link=4 Microdesc=1-2 Relay=2
required-client-protocols Cons=1-2 Desc=1-2 DirCache=1 HSDir=1 HSIntro=3 HSRend=1 Link=4 Microdesc=1-2 Relay=2
required-relay-protocols Cons=1 Desc=1 DirCache=1 HSDir=1 HSIntro=3 HSRend=1 Link=3-4 Microdesc=1 Relay=1-2
shared-rand-previous-value 1 X1ZxxMgtuCJ6Wu/4GAmN0/yJCRkbBGKN4n76akOybdY=
shared-rand-current-value 1 Ob9YcGpWWAd1V16YLNvBS0TixeBBwXI016zdPv9wvyg=
dir-source DA 5914F6630EE0CDD22429F22D408C63C287E138BF 59.110.54.10 59.110.54.10 7000 5000
contact auth0@test.test
vote-digest AE50FAA5B4F89C0E4B9ADB9C615F0FB3603B59CA
r RelayIndia DXSeJZ98quSCCgWjeQMgMwQ+sPg iN5GcezrPa6s/UCCyCqVQgWK9jQ 2021-02-21 00:00:35 147.139.37.130 25143 0
s Exit Fast Guard Running Stable Valid
v Tor 0.4.2.6
```

Fig. 3: Contents of the consensus file



Nickname	Advertised Bandwidth	Uptime	Country	IPv4	IPv6	Flags	Add. Flags	ORPort	DirPort	Type
● dizum (2)	18.08 MiB/s	26d 36m		45.66.33.45	-	● ⚡ ◐ ⇄ ○ ▮ ✓	▯	443	80	Relay
● Serge (1)	3.2 MiB/s	22d 9h		66.111.2.131	2610:1c0:0:5::131	● ⚡ ▮ ⇄ ○ ▮ ✓	▯	9001	9030	Relay
● moria1 (1)	500 KiB/s	36d 4h		128.31.0.34	-	● ⚡ ▮ ⇄ ○ ▮ ✓	⚠ ▯	9101	9131	Relay
● tor26 (1)	75 KiB/s	1h 29m		86.59.21.38	2001:858:2:2:aabb:0:563b:1526	● ⇄ ○ ▮ ✓	▯	443	80	Relay
● bastet (1)	50 KiB/s	11d 11h		204.13.164.118	2620:13:4000:6000::1000:118	● ⇄ ○ ▮ ✓	⚠ ▯	443	80	Relay
● maatuska (7)	50 KiB/s	42d 15h		171.25.193.9	2001:67c:289c::9	● ⇄ ○ ▮ ✓	▯	80	443	Relay
● dannenberg (1)	40 KiB/s	38d 20h		193.23.244.244	2001:678:558:1000::244	● ⇄ ○ ▮ ✓	▯	443	80	Relay
● Faravahar (1)	40 KiB/s	103d 9h		154.35.175.225	2607:8500:154::3	● ⇄ ○ ▮ ✓	▯	443	80	Relay
● gabelmoo (1)	40 KiB/s	60d 11h		131.188.40.189	2001:638:a000:4140::ffff:189	● ⇄ ○ ▮ ✓	▯	443	80	Relay
● longclaw (1)	38 KiB/s	5d 12h		199.58.81.140	-	● ⇄ ○ ▮ ✓		443	80	Relay
Total	22.09 MiB/s									

Fig. 4: Global Directory Server Information

At the same time, the Tor directory protocol also has shortcomings. Tor's 10 directory servers are "hard-coded" into clients as well as intermediate nodes[29]. As a result, when some authoritative directory servers are added and deleted, the program source code must be updated, which increases the cost of deployment and reduces the scalability.

Presence of malicious nodes: At present, there are roughly more than 6,500 distributed relay nodes and more than 1,700 bridge nodes in the entire network. Most of these nodes actively join and quit. Tor has no authentication mechanism for relay nodes, and there are a large number of unreliable relay nodes in the network. Attackers can deploy malicious nodes to steal and analyze the communication relationship in the link and destroy anonymity. And attackers can exploit existing vulnerabilities to attack these nodes, spread bots through various channels, infect a large number of hosts, and form botnets. Figure 5 illustrates the change in the number of relay nodes and bridge nodes from 202105 to 202107 and the node fluctuation [30].

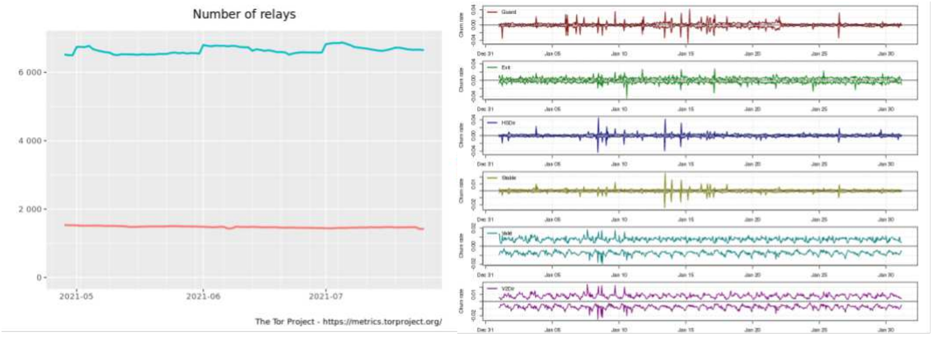


Fig. 5: Number of nodes and node fluctuation graph

2.3.3 Tor proposal

Since February 2018, the Tor community has submitted 48 proposals to the official Tor team[31], including improvements in security and performance. Among them, 23 proposals provide solutions for the security problems faced by Tor, 18 proposals provide solutions for Tor performance optimization, and 7 proposals provide improvements in other aspects. These proposals show that Tor still has corresponding problems in Dos attacks, censorship-resistant attacks, malicious node-in-the-middle attacks, and scalability.

At present, some scholars have used blockchain to realize anonymous communication between IoT devices[32]. The system divides the communication scope into two domains through centralized authentication and decentralized anonymous communication mechanisms. The zero-knowledge proof of identity is realized through the Merkle tree, the identity of administrator nodes is obfuscated and the association attack is resisted through aggregated signatures. Defects: IoT devices have high communication delay due to hardware performance and network environment limitations. And complex authentication and node management mechanisms are not suitable for large-scale users.

Qin Wang proposed a consortium system based on anonymous blockchain in 2021[33]. Since the data on the blockchain is open and transparent, a privacy system for protecting the blockchain-MAB is proposed. This system belongs to the application of privacy protection in the blockchain and lacks the versatility of anonymous communication systems.

3 Design

3.1 Threat Model

Consider an experienced attacker with sufficient computing power in the network who is trying to attack an anonymous communication system. The first consideration is the DOS attack. The attacker tries to perform a denial of service attack on the important nodes in the network so that it cannot serve other normal nodes, which makes the overall anonymous communication system unusable. This attack is generally effective against current anonymous communication systems such as Tor, and it is also the most difficult to defend against. B-Tor first adopts the blockchain distributed architecture, and each node is a peer node, which is a decentralized architecture. Secondly, the identity of the node is not fixed, the node will periodically run the reputation function (see 5.2) and redistribute the node function by calculating the reputation value. It makes it impossible for the attacker to select an important node to attack. If an ordinary node is attacked and paralyzed, the overall system will not be unable to run due to the offline or failure of a node. If an important node is paralyzed by an attack, the system can resume operation by redistributing node functions. Therefore, DOS attacks can be effectively prevented.

Second, consider malicious nodes joining. For the public anonymous communication system, malicious attackers join the system by pretending to be honest nodes. By passively collecting the traffic in the network, analyzing the time interval and the size of the data packets etc. It can determine whether there is a relationship between two users in the network. At the same time, the attacker can also hijack the traffic packets in the network, mark a group of traffic by discarding or modifying the traffic packets, and detect the traffic packets with this characteristic at a specific location, to analyze the correlation between users. B-Tor is a registered anonymous communication system. Nodes need to apply for registration to enter this anonymous communication system, which prevents malicious attacks from attackers. And in this way, the consensus file will not be exposed on the public network, which reduces the risk of being attacked by the exposure of distributed nodes in the system. At the same time, the authentication and node reputation management mechanism is introduced, which has a certain control effect on the behavior of joining nodes.

3.2 B-Tor Design Concept

Traditional distributed anonymous communication systems can be mainly divided into two categories, one is anonymous communication systems based on relay jumps, such as Tor, SGX-tor, shadow-walk, AP3, etc. One is the anonymous communication system based on the shuffling mechanism, such as loopix, riposte, Dissent, Atom[34]. The common problem is high network latency and weak defense against traffic analysis attacks by malicious nodes. The fundamental reason is the lack of censorship of malicious users and the identification of malicious users, and the complex network environment, the anonymous communication system bandwidth follows the barrel effect (the actual bandwidth is the maximum delay bandwidth in the node), which leads to the above-mentioned public anonymous communication System latency is high. At the same time, these anonymous communication systems have become criminal sanctuaries due to illegal abuse.

At present, when legitimate users use the Internet, they do not need to use anonymous communication systems in most cases. However, in special cases, legitimate users need to use anonymous communication systems to protect the legitimate behavior of individuals. However, it is blocked by many countries due to the above-mentioned hazards. In addition, the performance of network nodes is uneven, resulting in a large network delay. Therefore, the user's anonymity needs cannot be well satisfied. For example, the following legitimate anonymous requirements: (1) Anonymous reporting: users prevent malicious personnel from retaliation, and do not want their behavior to be discovered and tracked by others. (2) Anonymous voting: Users vote according to their wishes and do not want to be discovered by other users. (3) Anonymous award: users need to hide their identity to prevent others from hurting themselves maliciously due to jealousy. (4) Anonymous charity (5) Anonymous medical treatment: Patients are reluctant to reveal their identities to doctors and other groups of people. (6) Anonymous mailboxes of government departments, etc.

B-Tor was designed due to the above requirements and is a traceable anonymous communication system. The system requires users to trust an institution with strong credibility. The agency can be a national-level regulatory agency, such as Police Department, National Security Agency, etc. For example, in the above requirements: (1) Anonymous reporting, users trust the reporting agency. (2) Anonymous voting, users trust the voting institution. (3) To receive the award anonymously, the user trusts the awarding institution. (4) Anonymous charity, users trust charities. (5) Anonymous medical treatment, users trust medical institutions. (6) Government anonymous mailbox. Users trust the government and so on.

The above-mentioned trusted institutions cannot interfere with the normal behavior of users. Only when the user conducts illegal acts, the public trust agency can restore the registered identity of the anonymous user through the group signature according to the characteristics of this anonymous communication system. The public trust agency can restore the identity of the malicious user after obtaining the consent of the vast majority of legitimate users by broadcasting a retrospective request to the user. To achieve the purpose of traceability and supervision. From the perspective of game theory, if most legitimate users collude with public trust institutions to expose the identity of a legal user, then this behavior is detrimental to both legitimate users and trust institutions. Exposing an attacker or illegal user who has an impact on the system and society is beneficial to most users and institutions.

4 B-Tor System Architecture

Because of the shortcomings of the above-mentioned anonymous communication system Tor, this section introduces an anonymous communication system B-Tor based on the consortium blockchain. In this anonymous communication system, the bottom layer uses Tor's onion routing protocol and three-hop proxy communication mechanism to ensure the anonymity of users. At the same time, the upper layer uses the consortium blockchain technology in the blockchain technology, combined with the decentralization, non-tampering, and traceable characteristics of the blockchain, to solve the above problems.

4.1 Overview of B-Tor Architecture

This anonymous communication system is anonymous communication within the Hyperledger Fabric framework. The roles of the relay nodes in the original Tor network are divided so that the system operation is more stable and the node functions are clearer. This anonymous communication system needs to run on the Fabric distributed framework and consists of the following five components. The following describes each component and function.

Client node: The purpose of the client is to establish a link and initiate a session for the user, through which the user accesses the anonymous network. The main functions of the client are: send a registration request to the CA and obtain the consensus file for the first time, join the anonymous network, install and instantiate the chaincode, obtain the consensus file (steps 1-2 in Figure 6), and establish three based on the client's path selection algorithm. Hop nodes to build links (step 3 in Figure 6), etc.

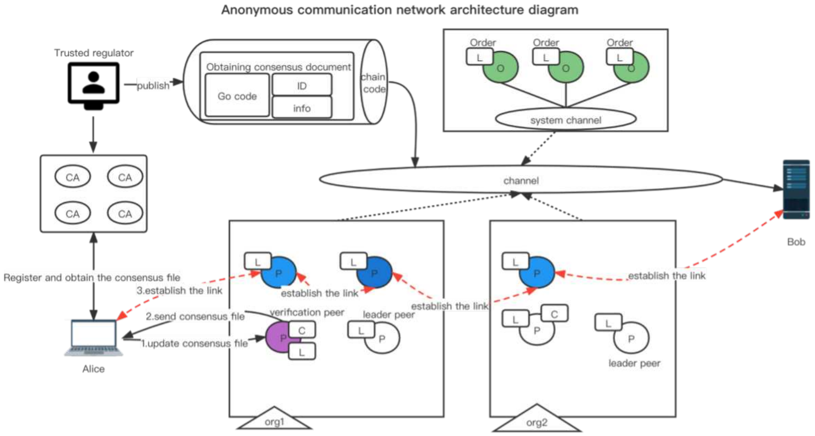


Fig. 6: System architecture diagram (where P stands for Peer, L stands for ledger, C stands for chaincode)

Relay node: The relay node is also called Peer. This node has the function of proxy forwarding and is a thoughtless node. At the same time, relay nodes are divided into four types: storage peer, verification peer, leader peer, and anchor peer. All types of peers have the functions of initiating registration requests to the CA (Certificate Authority), uploading and downloading distributed storage consensus files, and forwarding by three-hop agents. The verification peer has the unique function of verifying the chaincode. The verification peer is responsible for receiving the chaincode request submitted by

Table 1: Functions of peers

	Storage peers	Verification peers	Leader peers	Anchor peers
Register	✓	✓	✓	✓
Storage	✓	✓	✓	✓
Multi-hop Proxy	✓	✓	✓	✓
Verify Chaincode		✓		
Broadcast			✓	
Cross organization communication				✓

other peers and verifying the request. If the request is verified, it will be endorsed. The leader peer has a unique broadcast function, which is responsible for broadcasting the execution result of the order peer to other peers in its organization. The anchor peer has a unique cross-organization communication function, which can realize the communication between peers in different organizations in the same channel.

Order node: Can also be expressed as order peer. The function of the order node is to execute the transactions published by each node in the system, to execute the chaincode submitted by the client and the relay node, and to send the execution result to the leader peer of each organization.

Trust regulator: It is a credible supervisory organization whose main function is to publish chaincode and save the identity information of clients and nodes.

Channel: Responsible for connecting different nodes, only nodes that join the same channel can communicate with each other.

As an example: Assumption Alice and Bob want to communicate anonymously, where Alice is the user. Alice first registers with the CA to obtain the qualification to join the network and obtains a consensus file with anonymous network topology information. She can use the consensus file to establish a three-hop communication link to communicate with Bob. The intermediate node of the communication is a distributed network within the Fabric framework node. When the network topology changes, Alice executes the chaincode (smart contract) published in the channel to obtain the latest network topology (consensus file). Among them, when the chaincode is executed, it will be verified by the verification peer, and the consensus file can be obtained after the verification.

4.2 Running process

The overall framework of B-Tor is introduced in 4.1, and each function of the anonymous communication system is introduced one by one. Aiming at the network topology and communication mode of the anonymous communication system B-Tor, it is proposed that the construction of the anonymous communication system mainly needs to solve three problems: 1. How to join the nodes in the network. 2. How to build a link for communication. 3. How to update the consensus file when the network topology changes.

4.2.1 Registration and initialization

There are different participants in the B-Tor anonymous communication system, including relay nodes, client nodes, order nodes, etc. Nodes need to register their identity to join the network. Each node participating in anonymous communication has a digital identity encapsulated in an x.509 digital certificate. These identities are important, they limit the access rights of nodes in the anonymous network, and whether to connect to the anonymous network. B-Tor CA (CA for short) is a certificate authority that manages the identity of network nodes. It has the following functions: identity registration, issuance of certificates, and revocation of certificates.

Identity registration and certificate issuance:

(1) Register the boot ID. First, the node runs the B-Tor program and sends a registration request to the CA by constructing a boot identification command. The registration command will store the registration ECert(certificate) , the corresponding private key, and the certificate file PEM obtained from the CA request in the identity management directory MSP (member service provider) of the node.

(2) Register a new identity. Before registering a new identity, the CA checks the node. Three main aspects are checked. 1) Check whether the registered identity belongs to the corresponding organization. For example, if peer0 belongs to org1, then its registered identity must be peer0.xxx.org1.com, and the identity of other organizations cannot be registered. 2) Check whether the node has a boot ID, and check the node identity so that it cannot register a node that does not belong to its own identity. For example, client nodes cannot be registered as relay nodes or order nodes. 3) Check whether the node has been registered, and query the node's historical information and node-related attributes. If the node is registered, the check fails. A node that satisfies all of the above conditions can register an identity with attributes such as registration ID.

(3)Registration password. The node sets a registration password, and the system provides the registration ID and registration password to other nodes that have passed the registration for authentication between nodes during communication.

(4)Distribute consensus file. When the node is successfully registered through the above process, the CA sends the first consensus file. At this point, the new user completes registration and joins the network.

A consensus file contains three attributes: node ID, consensus file ID, and consensus file update time, of which the node ID contains 10 sub-attributes. The structure is shown in Figure 7.

Revoking a certificate: When the node information faces the risk of leakage or malicious behavior, the identity or certificate can be revoked. Revoking an identity will revoke all certificates owned by that identity and will also prevent that identity from obtaining any new certificates. Revoking a certificate will invalidate a single certificate. And generate a certificate revocation list after the revocation is complete. All requests by the CA server to receive a node whose identity has been revoked will be rejected.

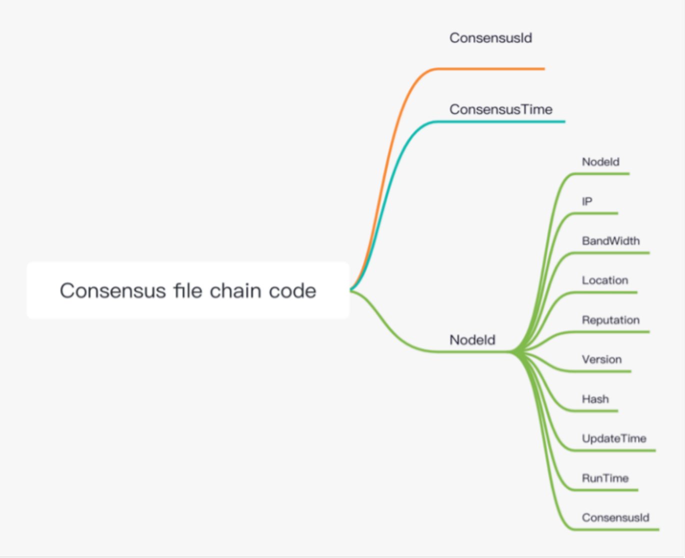


Fig. 7: Consensus file structure

4.2.2 Link Establish

Through the above process, the user completes the registration and obtains a consensus file with a description of the network topology. This section describes that the user obtains the IP address, bandwidth, and other information of the relay node in the system by parsing the consensus file (introduced in 5.1). Then start to establish a three-hop link. (1) Select the node with the highest reputation value (reputation value calculation is introduced in 5.2) as the first hop entry node Peer1;

(2)The client first sends a link establishment request to Peer1. After Peer1 verifies the legitimacy of the client, it will generate a pair of keys, the public key pubkey_Peer1_Client and the private key prikey_Peer1_Client. Then send the public key pubkey_Peer1_Client back to the client (so far, the client has successfully established its communication link with Peer1);

(3) The client selects a relay node Peer2 with a relatively high reputation value from the obtained consensus file, and sends a data packet to Peer1: use pubkey_Peer1_Client to encrypt the address of Peer2;

(4) After Peer1 receives the data packet, it uses its private key prikey_Peer1_Client to unpack the data packet, and finds that it is a request to establish a link between itself and another server, Peer2, then Peer1 repeats (2) to establish a link with Peer2, and Peer2 The returned public key pubkey_Peer1_Peer2 of the link between Peer1 and Peer2 is returned to the client;

The client repeats steps (3) and (4) to establish a communication link between Peer2 and Peer3, and receives the public key pubkey_Peer2_Peer3 of the link between Peer2 and Peer3;

So far, the link between the client and the three relay servers has been successfully established, and the client has three public keys: pubkey_Client_Peer1, pubkey_Peer1_Peer2, and pubkey_Peer2_Peer3. At this time, the client communicates anonymously through the three-hop proxy.

4.2.3 Network topology update

The network topology changes due to the addition and departure of relay nodes in the network. For the client and other relay nodes in the network to correctly obtain the network topology, the consensus file needs to be updated. Tor uses a directory server to centrally generate consensus files so that clients can obtain new network topologies. The client accesses the directory server through HTTP request to obtain the consensus file, which makes the Tor directory server centralization problem. To enhance security and resistance to censorship, we introduce a method for issuing consensus file through blockchain consensus.

The following specifically describes how to update the network topology through these four types of nodes. In 4.1, it is mentioned that the relay node Peer is divided into four types: storage peer, verification peer, leader peer, and anchor peer.

For example, when Peer0 joins the network, the network topology is updated.

1)The Peer0 node registers and joins the anonymous network through the joining method described in Section 4.2.1.

2)The Peer0 node broadcasts its node information to the network and records it in the blockchain. Specifically, first Peer0 joins the channel and obtains the chaincode in the channel. The function of the chaincode is to upload its attribute information to the blockchain according to the specified format and broadcast it to the network.

Peer0 first constructs the identity information and sends a request to execute the chaincode to the verification peer (shown in Figure 8, step 1). The verification peer checks its identity and simulates the execution of the chaincode. When the verification node checks that the node is a valid network member node and the chaincode simulation is executed correctly, it will sign the result and send it to Peer0 (shown in Figure 8, step 2).

After Peer0 gets the signature of the verification peer, it will send the signature to the order node and request to execute the chaincode. (shown in Figure 8, step 3).

The order peer does not verify the content of the chaincode, directly executes the chaincode requested by Peer0, and sorts the execution results. After sorting, the information of Peer0 is packaged into blocks and linked to the blockchain. (Shown in Figure 8, step 4)

The order node broadcasts Peer0 information to the leader peers of each organization. (The internal broadcast of org1 shown in Figure 8, step 5)

The leader peer of each organization is responsible for synchronizing the broadcast to every node in the organization. Each peer in the organization will record Peer0's information in the local ledger, complete a new peer joining the network and record the node joining information in the blockchain. (shown in Figure 8, step 6)

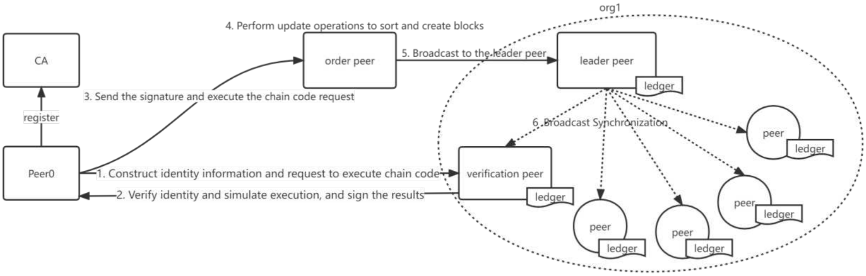

org1
4. Perform update operations to sort and create blocks
CA
order peer
5. Broadcast to the leader peer
leader peer
ledger
3. Send the signature and execute the chain code request
register
6. Broadcast Synchronization
peer
ledger
Peer0
1. Construct identity information and request to execute chain code
verification peer
2. Verify identity and simulate execution, and sign the results
ledger
peer
ledger
peer
ledger
peer
ledger

Fig. 8: Peer0 uploads information and issues consensus

3) Peer0 will periodically perform step (2) to synchronize its information to the system and broadcast the information to other peer in the network. The time set by this system is 10 minutes (which can be adjusted later), and the peer will automatically run the chaincode for updating its information regularly. After execution, each peer gets the latest information of Peer0 and updates the local consensus file.

4) If the Peer0 leaves or fails at a certain time, the chaincode cannot be executed regularly, so that the timestamp attribute in the Peer0 cannot be updated, so the timestamp information of the Peer0 in the local records of each peer is not synchronized with the current time. According to the node management and incentive mechanism to be introduced in 5.2, such nodes will not be selected as available nodes and will be automatically eliminated as the system runs.

Through the above method, each node can obtain a consensus file without using a directory server, and the network topology can be updated more conveniently. B-Tor does not have a centralized architecture, making the system more secure.

5 System optimization

5.1 Anonymous registration and traceability

Register anonymously:

The fourth section introduces that users need to register and manage the identity of network participants through B-Tor CA, but the way of registration will affect the anonymity of users. Aiming at the above problems, this paper proposes a node anonymization mechanism based on group signature. This registration mechanism adds a privacy protection method to the node authentication module in Hyperledger Fabric to ensure the anonymity of users and the security of the system.

The registration module consists of 5 parts, including RTCA (Root Certificate Authority), Fabric-CA cluster, B-Tor client node, relay node, and supervisor. The authentication part consists of a server cluster. The CA Server node is constructed in a tree structure, which includes a core RTCA node and multiple middleware nodes (Fabric-CA Intermediate Server). As shown in Figure 9. Node anonymous registration is mainly divided into 6 steps:

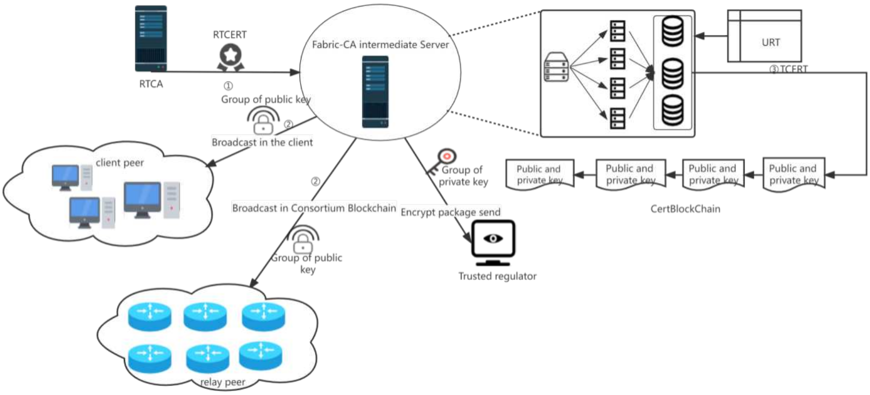


Fig. 9: Client and intermediate node registration architecture diagram

1) Root certificate generation:

When a consortium blockchain is being created, members of the consortium blockchain designate a node as a CA node through a configuration file. Then select a trusted third-party proxy certificate issuing agency RTCA (Root Certificate Authority) to generate RTCERT (Root Certificate). RTCERT is the root digital certificate of the entire network in B-Tor. The agency issues a sub-root certificate to the Fabric-CA intermediate Server according to RTCERT and writes the sub-root certificate into the configuration file when the consortium blockchain is created. As shown in ① in Figure 9.

2) Group public-private key generation.

After the consortium blockchain is started, CA nodes execute the generation of group public key and group private key for group signature, package the group public key information into a transaction, and then broadcast the transaction within this consortium blockchain, as shown in Figure 9 (②). After the nodes reach consensus, the transaction is uploaded to the certificate blockchain (CertBlockChain) as a genesis block. At the same time, the group private key information is encrypted and packaged into a transaction, and the recipient of the transaction is the supervisory node of this federated chain. After that, the broadcast consensus is made and the transaction is uploaded into CertBlockChain, as shown in Figure 9 (3).

3)Registration certificate issuance.

For the node that needs to join the consortium blockchain, the node initiates a registration transaction to the CA node, which contains the node's public key and necessary identity information; the CA node issues an access certificate ECERT (Enrollment Certificate) to the node after verifying the identity information it provides. At the same time as dispatching the ECERT, the CA issues a group certificate GCERT (Group Certificate) to the node. In this system, all nodes need to apply for the unique identity ECERT. both ECERT and GCERT are generated based on the node public key, and this process is shown in Figure 10, 1-3.

4)Issuance of consensus file.

The node that is successfully registered in the consortium blockchain provides the ECERT of the node and initiates the operation of obtaining consensus file to the CA node, which generates the TCERT (Transaction Certificate) of the pair by verifying the ECERT and issues the consensus file and TCERT to the corresponding node. Nodes can apply for TCERTs in advance when no transactions are made and can apply for multiple TCERTs in bulk. this process is shown in Figure 10, 4-5.

5) Transaction certificate up-chaining:

After the node has applied for TCERT to the CA node, the CA node needs to package the application into the transaction and up-chain the transaction to CertBlockChain to block-chain the consensus file and certificate dispatching for subsequent finding and supervision. The process is shown as ③ in Figure 9.

6) User and node information maintenance.

CA needs to maintain a URT (UserRegisterTable). When CA completes the registration of a node or abolishes a node authority according to the corresponding conditions, it needs to update the URT and keep the user data in the URT in a real-time updated state.

Traceability:

Nodes publish cross-chain transactions within the consortium blockchain and need to set a flag bit in the transaction to identify the cross-chain transaction. At the same time, it uses GCERT to sign the transaction and generate a group signature. Finally, the signed transaction is broadcasted within the consortium blockchain.

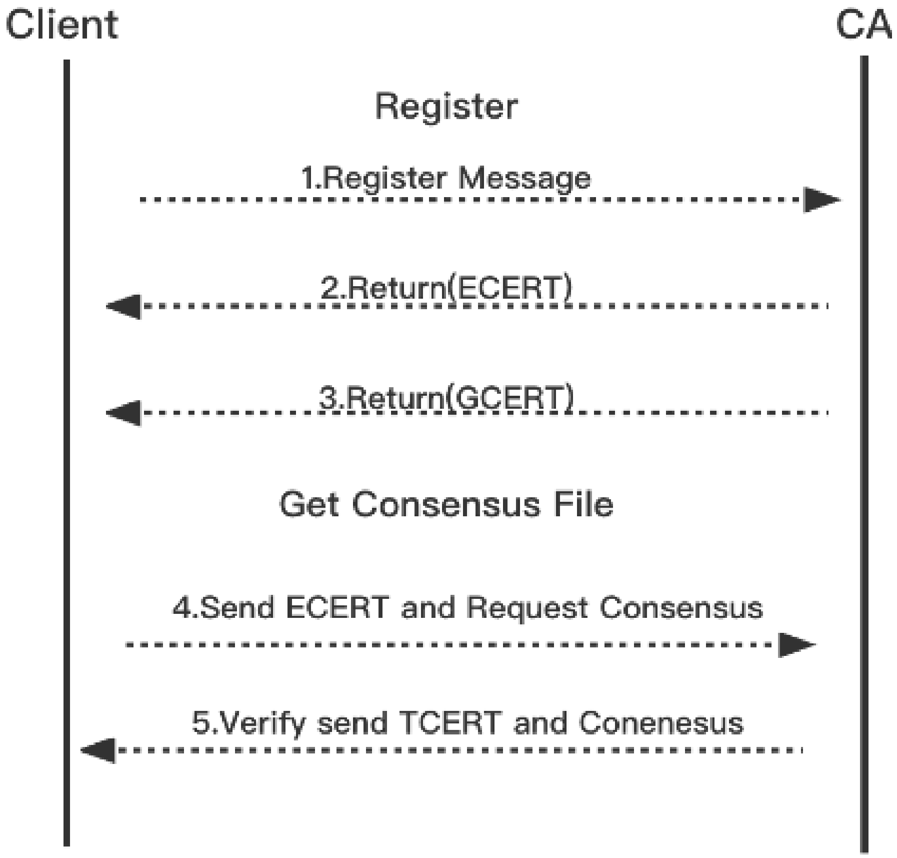


Fig. 10: Flow chart of user registration to CA

For nodes that have issued registration requests, CA or supervisory nodes trace the group signature of such transactions through the group private key to obtain the identity of the signer, to achieve the supervision of the de-anonymization of the members of the consortium blockchain and ensure that the identity of the signer of such transactions is not known to other members of the anonymous network.

At the same time, the sub root certificate and CA node information generated by RTCA, a third-party organization, will be packaged into a transaction and broadcast to the whole consortium blockchain network to finally reach a consensus. the generation of group public-private key by CA is automatically triggered by the configuration file when initializing the consortium blockchain. When CA generates GCERT for a node, it needs to bind ECERT and group public key. And GCERT contains the identity information of the node, when the regulator needs to de-anonymize the node, the identity of the node can be recovered by the group private key.

Through the above-mentioned group signature, the registered users can be anonymized and their identities can be protected. The introduction of the registration mechanism increases the cost of malicious node attacks to some extent and increases the trustworthiness of relay nodes to avoid zombie nodes from disrupting the network. At the same time, combining the characteristics of group signature and blockchain, the identity of the evil node can be recovered through the supervisor when there exists a node to do evil, which achieves the purpose of traceability for malicious users. This traceable anonymous communication system, to a certain extent, prevents the abuse of anonymous networks from bringing social insecurity.

5.2 Node Management and Incentive Mechanism

Node management:

It is mentioned in the system architecture that different relay nodes have different attributes and functions so there will be the following disadvantages compared to the Tor relay node which has the same function. The verification peer is responsible for the overall system transaction validation, the leader peer is responsible for the dissemination of the consensus file down in the system, and the anchor peer is responsible for the cross-channel communication of the system. This solidification of node functions can easily lead to insecurity of the system and the possibility of malicious nodes committing mischief in the long run. Therefore, B-TOR introduces a node management mechanism that calculates the reputation value of each node and assigns and adjusts the node functions according to the different reputation values.

First, after each relay node installs the B-TOR program and joins the system, it automatically deploys a Reputationfunc smart contract that calculates the reputation value and provides an API for external calls to this smart contract, which is forced to run in the anonymous communication system and cannot be modified by the user in terms of execution process and sequence. The Reputation function generates its Reputation value, which is written as an attribute of the node in the consensus file and is available to other nodes in the system.

By judging the reputation value, the function of each node is assigned and whether it can join this anonymous network or not. The overall flow of the system is shown in Figure 11. When the reputation value is invalid, the node will be denied access to the network. When the reputation value is valid, the identity is reassigned. At the same time, nodes that are denied access to the network can request access to the network again and recalculate the reputation value. According to the behavioral attributes of nodes, they can be divided into irrational and rational behaviors. More irrational behavior means that this node is more unreliable and unstable, and similarly, the higher rational behavior of a node means that the node is more stable and trustworthy. Tables 2 and 3 specify the rational and irrational behaviors.

Firstly, the node needs to execute the chaincode and request the verification peer to verify it (as shown by 4.2.3 Execution flow). Specifically, the node sends

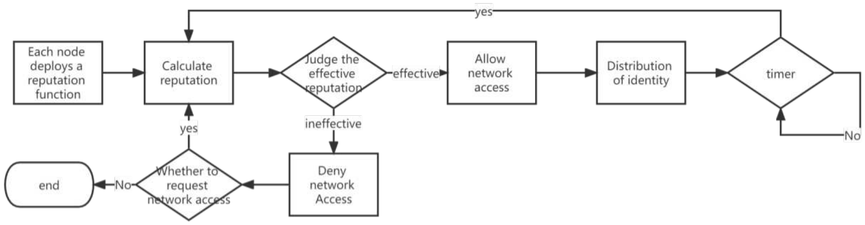


Fig. 11: Node reputation generation process

Table 2: Irrational behavior

Property	Property Description
Cheat	When a node communicates with other nodes, it deceives the trust of the other party and provides false transaction information.
Slander	When a node communicates with other nodes, it attacks other nodes in various ways.
Assume another's name	Nodes impersonate other nodes and perform the functions of other nodes.
Lurk	The node does not act and does not perform the function of its own node.
Conspire	Nodes collude with other nodes to improve each other's reputation, or collude to attack the system.
Reentry	Nodes frequently access the system, or re-enter the system by changing their identities.

Table 3: Rational behavior

Property	Property Description
Bandwidth	The bandwidth of nodes is significantly higher than that of ordinary nodes.
Online time	The online time of the node is longer, and the communication with other nodes is more frequent.
Provide service time	Nodes provide more proxy communication and high data integrity.
Historical reputation	Rich node history and high reputation values
Computing power	Nodes take less time to execute transactions and publish them on the blockchain.
Number of local ledgers	The local ledger capacity of the node is large, and the historical data is stored more.

the execution content to the verification peer, at which point the verification peer verifies the authenticity of this Peer node content. The verification node calls the Reputationfunc smart contract API of the Peer node from outside and gets a copy of the execution result. The behavior of the Peer node is verified by judging whether it is consistent with the information submitted by the Peer node. Based on the behavior verification node will further modify the reputation value submitted by the Peer node to ensure the authenticity of the node information.

Reputation calculation function Reputationfunc:

Reputation calculation function will calculate reputation value Reputation based on the above user behavior, rational behavior by executing the function Contribution() to obtain rational behavior value. Irrational behavior is calculated by executing the destroy () function to obtain the irrational behavior value γ.The Reputation value Reputationj of node Peer j is calculated by the function.

$$\text{Reputation }_j = \sum_{\text{n}=0}^{k} \max\left\{\sum_{i=0}^{6} \alpha_i^j, -\sum_{i=0}^{6} \beta^{\gamma_i^j}\right\} \tag{1}$$

Where α_i^j denotes the ith rational behavior value of Peer j. γ_i^j denotes the ith irrational behavior value of Peer j.

where n is the number of times Peer j performs the reputation function in the system. β is the damage degree factor, and β can be adjusted according to the later need. If a node has irrational behavior, the value of irrational behavior of the node will show an exponential increase, which makes the overall reputation value of the node drop, and when the reputation value of the node drops to 0, this node cannot be used. Whenever the system performs a reputation function, its value will be stored in the blockchain, which is called historical reputation, and each node keeps up to 10 recent historical reputation values.

Incentive mechanism:

Also in distributed networks, intermediate nodes are responsible for message forwarding, and artificially deploying a large number of nodes will increase the cost of running anonymous networks. An incentive mechanism needs to be considered to make relay nodes join as volunteers voluntarily. This is specifically achieved through the above-mentioned node reputation, which is obtained by the node and does not disappear when the node leaves the system but is permanently stored in the blockchain network. Client nodes can communicate anonymously based on their reputation, and when a relay node wants to convert to a client node, it can obtain the reputation of the relay node belonging to itself in the blockchain-based on its private key (refer to the Bitcoin wallet mechanism here, where Bitcoin users can obtain bitcoins in the wallet based on their private key). The credibility translates into credit for using the B-Tor anonymous communication system, the higher the credit the longer the time to communicate anonymously. This allows client nodes that need anonymity to voluntarily join the relay nodes. At the same time, users consider the impact of reputational value, which discourages irrational behavior.

6 Evaluation

For the experimental testing of the above architecture, the operating system used was Ubuntu 1604, the CPU was AMD Ryzen 9 5900X, the RAM was 16G, the SSD was 100G, and the default consortium blockchain framework was Hyperledger Fabric v2.0.0. The test was opened in two organizations with a total of 10 test nodes, and the experiment first deployed Hyperledger Fabric consortium blockchain environment and deploy B-Tor to that environment.

By writing chaincode in go language combined with fabric-go-sdk, we implement operations related to registering nodes to join the network and obtaining distributed node information in the B-Tor system. This includes 1) initialize the consensus file. 2) add nodes. 3) update node information. 4) query node information by node ID. 5) get a new consensus file, etc.

After the client nodes and intermediate nodes deploy the B-Tor program, they will execute the internal chaincode, and the initialization operation will be performed when the program starts, and the chaincode will be installed. This action will check the user and client identity, and the chaincode can be executed if the identity requirements are met. When the program is initially run, the consensus file is initialized, and it is necessary to execute initConsensus().

Experiments related to client access to the consensus file are also conducted. The number of times the consensus file is successfully acquired is counted by 10 client nodes continuously acquiring the consensus file. Figure 12 shows that the B-Tor client continuously acquires the consensus file in 600s time, and the total number of successful acquisitions is 790629 in 600s. The delay of acquiring consensus files for 10 clients is also tested, and the average is around 0.1s. The experiments show that the B-Tor anonymous communication system has significantly shorter latency and a higher success rate in acquiring consensus files by clients. It is suitable for large-scale distributed networks. We used the official Tor network simulation tool shadow[35]with the Tor performance simulation tool tornettools to perform simulation experiments[36]. The experiments were conducted using the real official Tor data[37]and scaled down to 0.005% of the real network. Three directory servers and 6746 intermediate nodes were used and the simulation duration was 600 s. This experiment makes comparisons in terms of link communication round trip time, consensus file acquisition time, data transfer time, and link establishment time.

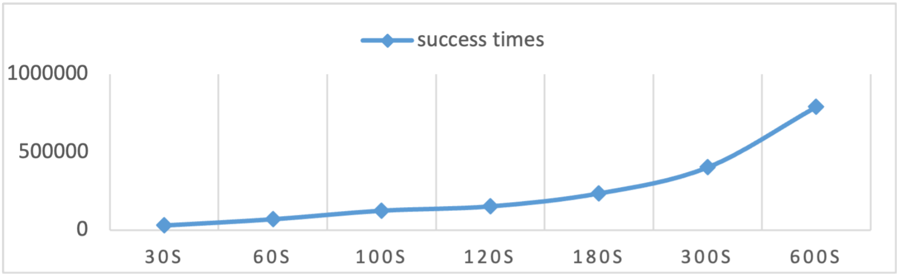


Fig. 12: Number of successful consensus file acquisition by B-Tor

The average time for Public-Tor to build a three-hop link and communicate back and forth is shown in Figure 13. Public-Tor takes 1∼2 s, and a few round-trip links take more than 5 s. B-Tor takes about 1 s, and the longest time is 2.2 s. It is better than public-tor in terms of time and stability.

The average time to obtain a consensus file is 2.2s for public-Tor(show in Figure 14), and 1.2s for B-Tor. the performance of the network can be judged by the time to obtain a consensus file, and the time to obtain consensus file is also the main factor to determine the network latency. Compared to public-Tor, this anonymous communication system has lower latency.

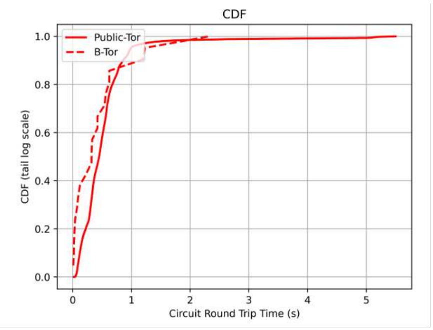


Fig. 13: CDF diagram of link communication round trip time

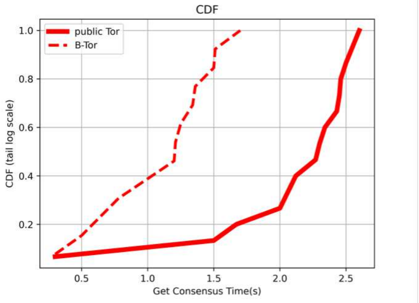


Fig. 14: CDF plot of average time to obtain consensus file

The transmission rate is an important indicator of anonymity network performance(show in Figure 15). This experiment compares public-Tor and B-Tor by sending a 1M packet at the same time and measuring the average time. public-Tor takes 4 5s on average and some packets are lost, B-Tor takes 2.5s on average and the transmission process is more stable than public-Tor.

Link establishment time directly affects user experience and is the main criterion for evaluating anonymous network performance. By obtaining the link establishment time several times, we obtain the relevant data and plot the link establishment time CDF(Cumulative distribution function)diagram in Figure 16. Compared with public-Tor, the link establishment time of B-Tor is significantly shorter, with 95% of nodes being established within 1 second. The average time reduction is one-third of that of public-Tor.

This system load was tested by Hyperledger Caliper[38] on the performance of the chaincode and the system load. We focus on the read consensus file function and thus test the read and write performance of the system. The test environment is an Ubuntu 1604 virtual machine with AMD Ryzen 9 5900X CPU and 16G RAM.

Table 4 shows the system resource consumption in the docker environment, using Peer0 node of organization 1 to test the system load, from which we can get the maximum system CPU load rate of 77.36%, the average CPU usage CPU 32.34%, the maximum number of memory used 129M, the average

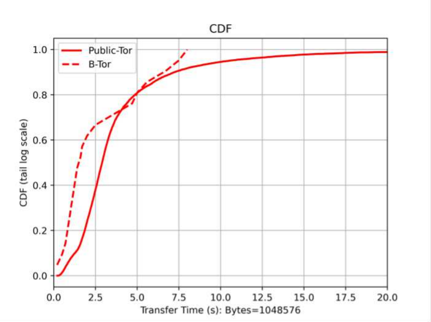


Fig. 15: CDF diagram of 1MB data transfer time

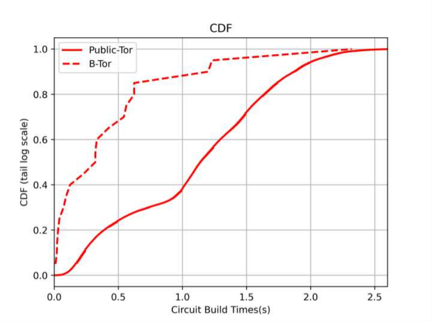


Fig. 16: CDF diagram of the one-way time to establish a link

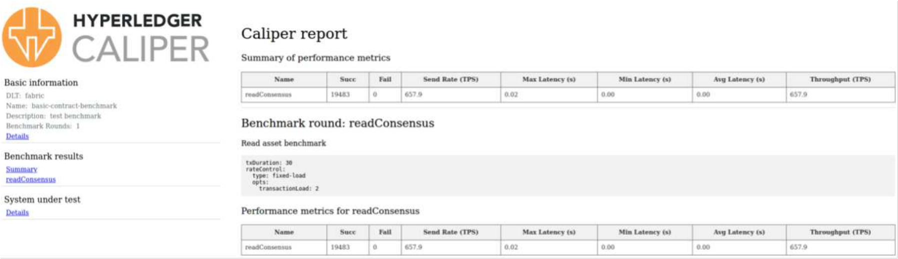
memory usage 128M, the entrance traffic 42M, the exit traffic 63M, the disk read and write information was 292Kb for writes and 108Kb for reads.

Also, the initial test used two clients, cyclic test 30s to get the test results, showing the read 19,483 times, the success rate of 100%. The send rate was 657.9 TPS and the system throughput was 657.9 TPS. Figure 17 shows some of the results of this experiment.

And for two clients, circular call to obtain consensus file chaincode for the 30s, Figure 17 gives the report results.

According to the above test method, the performance test is conducted by increasing the number of clients and changing the cyclic reading time. Specifically, 10 clients were used to cycle through the 60s, 100s, 120s, 180s, and 300s to see the system load. The system throughput was relatively stable during the 600s test time, reaching an average of 1312.9TPS and a maximum of 1352.8TPS. Figure 18 shows the results of this test.

From the experimental results we get the system overall increases the number of executed transactions as the test time increases. This system has a large throughput and sending rate and can remain relatively stable and unchanged during the long-time cyclic test.



Basic information

DLT: fabric
Name: basic-contract-benchmark
Description: test benchmark
Benchmark Rounds: 1
Details

Benchmark results

Summary
readConsensus

System under test

Details

Caliper report

Summary of performance metrics

Name	Succ	Fail	Send Rate (TPS)	Max Latency (s)	Min Latency (s)	Avg Latency (s)	Throughput (TPS)
readConsensus	19483	0	657.9	0.02	0.00	0.00	657.9

Benchmark round: readConsensus

Read asset benchmark

```
txDuration: 30
rateControl:
  type: fixed-load
  opts:
    transactionLoad: 2
```

Performance metrics for readConsensus

Name	Succ	Fail	Send Rate (TPS)	Max Latency (s)	Min Latency (s)	Avg Latency (s)	Throughput (TPS)
readConsensus	19483	0	657.9	0.02	0.00	0.00	657.9

Fig. 17: caliper system load report

Table 4: Hyperledger Caliper Partial Performance Values

Node	CPU%(max)	CPU%(avg)	Memory(max) [MB]	Memory(avg) [MB]	Traffic in [MB]	Traffic out [MB]	Disc wrtie [MB]	Disc Read [Kb]
Dev-Peer0-org1	46.02%	19.16	22.3	22.2	34.2	13.5	0.00	4.00
Dev-Peer1-org2	0.15	0.04	8.84	8.79	0.0399	0.0153	0.00	0.00
Peer0-org2	2.00	1.36	117	117	0.189	0.129	292	56.0
Order	0.78	0.15	31.7	30.9	0.101	0.195	280	136
Peer0-org1	77.36	32.34	129	128	42.1	63.0	292	108

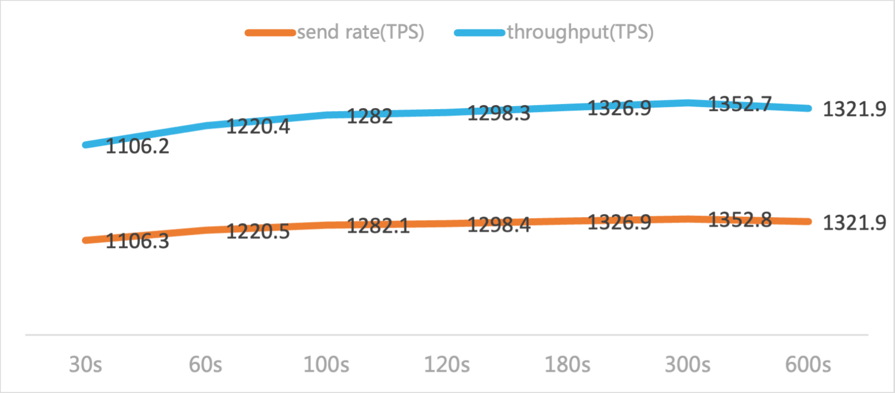


Fig. 18: Variation of system throughput at different times

7 Summary

With the continuous development of network security, data security and personal privacy security have been gradually concerned by the state and society. cryptographic algorithms ensure the security of data contained in Internet communication, and anonymous communication technology ensures the security of user privacy in Internet communication. tor is currently the most widely used open-source anonymous communication system, and with the use of tor, more and more people are studying its characteristics and security. In this paper, we analyze the Tor architecture, and deeply analyze the existence of centralization and other security issues in the Tor architecture. In addition, we apply the decentralized, tamper-evident and traceable features of blockchain to the Tor architecture to generate a consortium blockchain-based anonymous communication system B-Tor.

The design idea of this paper adopts the consortium blockchain as the underlying architecture of B-Tor. It is specifically implemented through Hyperledger Fabric, the most widely used in consortium blockchain. The consensus file of the whole anonymous communication system is stored through the Hyperledger of each peer node. B-Tor has the following features: protection of users' normal anonymous communication, verifiability of joined relay nodes, traceability of transactions against crimes, and distributed storage of consensus files to solve the problem of directory server centralization.

Section 1 of this paper illustrates the importance of the existence of anonymous communication systems and presents the problem of misuse of Tor to make it unusable for some users. Section 2 specifies the architecture of Tor and blockchain technology and introduces the current attacks and flaws faced by the Tor network. Section 3 gives the current threat model of anonymous communication systems in response to these flaws and describes the B-Tor design concept. Section 4 specifies the overall system model of B-Tor, as well as the functions and implementation methods of each module. It also explains the introduction of a new consensus file update and distribution method compared with traditional Tor, and the addition of node management functions to ensure more security during system operation. Section 5 presents the experiments and analysis of this system, and the advantage of B-Tor over Tor is the different way of obtaining consensus files. The first section of Section 5 demonstrates B-Tor's access to consensus files in the consortium blockchain and the related comparison tests. It also shows the advantages of high throughput and high performance compared to public blockchains due to the use of the Fabric consortium blockchain architecture. The second subsection demonstrates the load of the system when B-Tor is cycling through multiple users to obtain consensus files. Finally, the analysis results of this system are given.

This paper only makes a preliminary attempt for the anonymous communication system Tor combined with blockchain, and the next work needs to be done to improve and think about the following aspects: for how organizations and users join and exit the system in most consortium blockchains. This experiment is relatively simple, without large-scale users for testing, while the number of organizations joining the consortium blockchains is small, and the system performance needs further analysis.

The underlying source code of Tor and the mechanism of establishing and selecting links have not been modified, and the characteristics of Tor traffic still exist. Further research is needed on fingerprinting attacks and other ways to target traffic.

Regarding node management, a method to calculate reputation worthiness is introduced, and it is hoped that a more secure and stable node management algorithm can be added on this basis subsequently.

Acknowledgments. This research was supported by the National Defense Basic Scientific Research program of China, project number: JCKY2018602B015

Declarations

Ethical Approval and Consent to participate: Not applicable

Human and Animal Ethics: Not applicable

Consent for publication: Not applicable

Availability of supporting data: All data generated or analysed during this study are included in this published article

Competing interests: The authors declare that there is no conflict of interest regarding the publication of this paper.

Funding: Not applicable

Authors' contributions: Dawei xu and Jiaqi Gao write papers and do experiments. Liehuang Zhu and Feng Gao proposed and designed the key technologies of this paper. Yang Han and Jian Zhao collect experimental data and find information.

Acknowledgements: This research was supported by the National Defense Basic Scientific Research program of China, project number: JCKY2018602B015

Authors' information:

Dawei Xu, is a PhD student in School of Cyberspace Science and Technology, Beijing Institute of Technology. He engages in science research and education work in College of Cybersecurity, Changchun University. His current research interests include blockchain technology, anonymous communication, big data privacy protection, and machine learning. Email: xudw@ccu.edu.cn

Jiaqi Gao, is a graduate student majoring in cyberspace security, Changchun University. His research interests include blockchain technology, and anonymous communication. Email: judgegao06@gmail.com

Liehuang Zhu, is a Professor in School of Cyberspace Security, Beijing Institute of Technology. His current research interests include Cryptographic algorithm and security protocol, block chain technology, cloud computing security, big data privacy protection, artificial intelligence security, Internet of things security, heaven and earth integration network security. Email: liehuangz@bit.edu.cn

Feng Gao(gaofengbit@foxmail.com) received a Ph.D. degree in the School of Computer Science and Technology at Beijing Institute of Technology, China, in 2018. His research interests include blockchain and cybersecurity.

Yang Han, is currently a master student in the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China. Her research interests include blockchain technology, anonymous communication, and privacy protection. Email: y_han@outlook.com

Jian Zhao received PhD degree in Computer Science and Technology from Jilin University, Changchun, China, in 2012. Now he works at Changchun University. His current research interests include control, dynamics and fault diagnosis. Email: zhaojian@ccu.edu.cn

Appendix A

5.1 Description of abbreviations

CA:Certificate Authority
RTCA:Root Certificate Authority
RTCERT:Root Certificate
ECERT:Enrollment Certificate
GCERT:Group Certificate
TCERT:Transaction Certificate
URT :UserRegisterTable

References

[1] Isaak, J., Hanna, M.J.: User data privacy: Facebook, cambridge analytica, and privacy protection. Computer **51**(8), 56–59 (2018)

[2] Montieri, A., Ciuonzo, D., Aceto, G., Pescapé, A.: Anonymity services tor, i2p, jondonym: classifying in the dark (web). IEEE Transactions on Dependable and Secure Computing **17**(3), 662–675 (2018)

[3] Luo, J., Yang, M., Ling, Z., Wu, W., Gu, X.: Anonymous communication and darknet: A survey. Journal of Computer Research and Development **56**(1), 103 (2019)

[4] Karunanayake, I., Ahmed, N., Malaney, R., Islam, R., Jha, S.: Anonymity with tor: A survey on tor attacks. arXiv preprint arXiv:2009.13018 (2020)

[5] Nazah, S., Huda, S., Abawajy, J., Hassan, M.M.: Evolution of dark web threat analysis and detection: A systematic approach. IEEE Access **8**, 171796–171819 (2020)

[6] van der Bruggen, M., Blokland, A.: Profiling darkweb child sexual exploitation material forum members using longitudinal posting history data. Social Science Computer Review, 0894439321994894 (2021)

[7] Martin, J., Munksgaard, R., Coomber, R., Demant, J., Barratt, M.J.: Selling drugs on darkweb cryptomarkets: differentiated pathways, risks and rewards. The British Journal of Criminology **60**(3), 559–578 (2020)

[8] Nazah, S., Huda, S., Abawajy, J., Hassan, M.M.: Evolution of dark web threat analysis and detection: A systematic approach. IEEE Access **8**, 171796–171819 (2020)

[9] Alharbi, A., Faizan, M., Alosaimi, W., Alyami, H., Agrawal, A., Kumar, R., Khan, R.A.: Exploring the topological properties of the tor dark web. IEEE Access **9**, 21746–21758 (2021)

[10] Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. Technical report, Naval Research Lab Washington DC (2004)

[11] Iszaevich, G.E.W.: Distributed detection of tor directory authorities censorship in mexico. In: International Conference on Networks (2019)

[12] Haraty, R.A., Assi, M., Rahal, I.: A systematic review of anonymous communication systems. ICEIS (2), 211–220 (2017)

[13] Winter, P., Edmundson, A., Roberts, L.M., Dutkowska-Żuk, A., Chetty, M., Feamster, N.: How do tor users interact with onion services? In: 27th USENIX Security Symposium (USENIX Security 18), pp. 411–428 (2018)

[14] Nakamoto, S., Bitcoin, A.: A peer-to-peer electronic cash system. Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf **4** (2008)

[15] Zheng, Z., Xie, S., Dai, H.-N., Chen, X., Wang, H.: Blockchain challenges and opportunities: A survey. International Journal of Web and Grid Services **14**(4), 352–375 (2018)

[16] Cambiaso, E., Vaccari, I., Patti, L., Aiello, M.: Darknet security: A categorization of attacks to the tor network. In: ITASEC, pp. 1–12 (2019)

[17] Jansen, R., Tschorsch, F., Johnson, A., Scheuermann, B.: The sniper attack: Anonymously deanonymizing and disabling the tor network. Technical report, Office of Naval Research Arlington VA (2014)

[18] Jansen, R., Vaidya, T., Sherr, M.: Point break: A study of bandwidth {Denial-of-Service} attacks against tor. In: 28th USENIX Security Symposium (USENIX Security 19), pp. 1823–1840 (2019)

[19] Mane, Y.D., Khot, U.P.: An efficient technique to detect slow rate ddos attack from a private tor network. International Journal of Electronic Security and Digital Forensics **13**(1), 88–104 (2021)

[20] Wang, R., Yang, Z., Li, K., Chen, C., Chen, Y.: Research on tor-based anonymous ddos attack. In: MATEC Web of Conferences, vol. 336 (2021). EDP Sciences

[21] Basyoni, L., Fetais, N., Erbad, A., Mohamed, A., Guizani, M.: Traffic analysis attacks on tor: a survey. In: 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT), pp. 183–188 (2020). IEEE

[22] Xueliang, S., Anxin, H., Xiapu, L., Yi, X.: Webpage fingerprinting identification on tor: A survey. Journal of Computer Research and Development **58**(8), 1773 (2021)

[23] Lashkari, A.H., Draper-Gil, G., Mamun, M.S.I., Ghorbani, A.A.: Characterization of tor traffic using time based features. In: ICISSp, pp. 253–262 (2017)

[24] Platzer, F., Schäfer, M., Steinebach, M.: Critical traffic analysis on the tor network. In: Proceedings of the 15th International Conference on Availability, Reliability and Security, pp. 1–10 (2020)

[25] Sanatinia, A., Noubir, G.: Off-path man-in-the-middle attack on tor hidden services. New England Security Day, NESD (2017)

[26] Winter, P., Ensafi, R., Loesing, K., Feamster, N.: Identifying and characterizing sybils in the tor network. In: 25th USENIX Security Symposium (USENIX Security 16), pp. 1169–1185 (2016)

[27] Pries, R., Yu, W., Fu, X., Zhao, W.: A new replay attack against anonymous communication networks. In: 2008 IEEE International Conference on Communications, pp. 1578–1582 (2008). IEEE

[28] torproject: TorMetrics. [EB/OL]. https://metrics.torproject.org/networksize.html Accessed April 4, 2022

[29] onbasca: Tor directory protocol, version 3. [EB/OL]. https://onbasca.readthedocs.io/en/latest/dir-spec.html Accessed December 10, 2021

[30] torproject: TorMetrics. [EB/OL]. https://metrics.torproject.org/rs.html#search/flag:Authority Accessed March 6, 2022

[31] torproject: proposals. [EB/OL]. https://gitweb.torproject.org/torspec.git/tree/proposals Accessed March 6, 2022

[32] Lu, Y.: A p2p anonymous communication scheme in iot based on blockchain. International Journal of Network Security **23**(1), 49–56 (2021)

[33] Wang, Q., Chen, S., Xiang, Y.: Anonymous blockchain-based system for consortium. ACM Transactions on Management Information Systems (TMIS) **12**(3), 1–25 (2021)

[34] Shen, T., Jiang, J., Jiang, Y., Chen, X., Qi, J., Zhao, S., Zhang, F., Luo, X., Cui, H.: Daenet: making strong anonymity scale in a fully decentralized network. IEEE Transactions on Dependable and Secure Computing (2021)

[35] Jansen, R.: shadow. [EB/OL]. https://github.com/shadow/shadow. Accessed January 28, 2022

[36] Jansen, R., Tracey, J., Goldberg, I.: Once is never enough: Foundations for sound statistical inference in tor network experimentation. In: 30th USENIX Security Symposium (USENIX Security 21), pp. 3415–3432 (2021)

[37] torproject: collector. [EB/OL]. https://collector.torproject.org/archive/ Accessed February 8, 2022

[38] LinuxFoundationProject: HyperledgerCaliper. [EB/OL]. https://www.hyperledger.org/use/caliper Accessed February 8, 2022