ACS: An Efficient Messaging System with Strong Tracking-Resistance

Zhefeng Nan[1,2], Changbo Tian[1,2(✉)], Yafei Sang[1,2], and Guangze Zhao[3]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
{nanzhefeng,sangyafei}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China
tianchangbo@iie.ac.cn
[3] Chinese Asset Cybersecurity Technology CO., Ltd, Beijing 100041, China

Abstract. The increasingly rampant network monitoring and tracing bring a huge challenge on the privacy protection, because even if the message data is encrypted, the communication privacy is difficult to be hidden. Existing anonymous systems sacrifice anonymity for efficient communication, or vice versa. In this paper, we present *ACS*, an efficient messaging system which leverages a two-layer framework to provide tracking-resistance. The first layer is the *entry layer*, which consists of *entry servers* to relay messages. The second layer is the *exchange layer*, which consists of *exchange servers* to exchange messages. Users divide its message into different shares and send each share to exchange server via a randomly chosen entry server. Users only provide their pseudonyms to exchange servers for message exchange. Then, entry servers have no information about the message exchange, and exchange servers have no information about users' identities. The exchange servers also provide message storage service in case that the receiver of these messages are offline, in which way, the communication becomes more simple and flexible. The experimental results show that our proposed system guarantees the strong tracking-resistance and high communication efficiency.

Keywords: Anti-tracking network · Anonymous communication · Message segmentation · Offline messaging · Privacy protection · Cyber security

1 Introduction

The security of communication privacy attracts more and more attention due to the disclosure of extensive mass surveillance programs [1–4], especially, some network surveillance and censorhip programs are participated or dominated by

Supported by the National Key Research and Development Program of China under Grant No.2019YFB1005205.

H. Gao et al. (Eds.): CollaborateCom 2022, LNICST 461, pp. 55–74, 2022.
https://doi.org/10.1007/978-3-031-24386-8_4

the state. The protection of communication privacy and users' identities in the Internet has become an increasingly important security requirement.

To address this problem, anonymous network has been proposed as the countermeasure to fight against the network surveillance and censorship [5]. Tor, a practical manifestation of Onion-routing, has become the most popular anonymous network. Unfortunately, the Onion-routing based anonymous networks are susceptible to traffic analysis attack [6–8] by an adversary that can monitor or tamper with network traffic between nodes. Mix-net [9] based anonymous networks are message-oriented systems which confuse the network traffic through mix nodes to improve anonymity. Due to the high latency and computation overhead of mixing, they can hardly accommodate real-time and high bandwidth communication. DC-net [10,11] based anonymous networks allow multiple parties to implement a broadcast channel to prevent each party from distinguishing the message sender and receiver. However, they sacrifice the bandwidth for the anonymity.

With the development of network technologies, the techniques of network surveillance and censorship become more diverse and sophisticated. And the adversary may have a global view of network and continuously monitor the Internet backbone. Furthermore, the adversary may take the active attack to crack down the anonymous system, such as the DDoS attack or information tampering. So, A successful system needs to resist powerful active and passive attacks, and provide efficient and secure communication.

In this paper, we propose an anti-tracking messaging system called ACS. ACS system contains two kinds of servers: entry server and exchange server which construct the two-layer framework of ACS. The entry server is only used to relay the messages between the user and exchange layer. In this way, the entry server breaks the direct relationship between the user and exchange server. Hence, each user only needs to provide its pseudonym to exchange server for message exchange. The message exchange is implemented only according to the users' pseudonyms.

To avoid information leakage, each sender divides its message into different shares and sends each message share to one exchange server for message exchange. Only received all the message shares from the exchange servers, the receiver can reconstruct the original message. The entry server and exchange server can not get any valid information from each message share.

Each message share is delivered by a randomly chosen entry server to improve security and anti-tracking in network communication. The exchange server also stores the valid message shares if the receiver is offline. When the receivers access ACS system again, they can receive the message stored in the exchange servers. The offline messaging avoids the negotiation and confirmation between the communicating parties, which provides a more flexible transmission mechanism.

The contributions of this paper are outlined as follows:

– We design an anti-tracking messaging system (ACS), which divides the messaging process into message forwarding and message exchange through two-layer (entry layer and exchange layer) messaging framework. Entry layer hides the users' identities from the exchange layer and the exchange layer hides the

communication relationship from the entry server. ACS provides low-latency communication and high tracking-resistance.
- ACS system provides offline messaging which makes the communication more flexible. The communicating parties need no negotiation and confirmation with each other. Then, each user doesn't need to stay online in ACS system all the time to wait for the possible messaging process. Also, offline messaging makes the messaging process happen in different time, which improves the difficulty in traffic analysis of communication relationship.
- ACS system reduces the information leakage to the minimum level through message segmentation mechanism. The original message will be split into different shares, and each message share will be sent to different exchange server via different entry server. Only received all the message shares, the receiver can reconstruct the original message.

The rest of the paper proceeds as follows. In Sect. 2, we introduce the noticeable anonymous networks and technologies. In Sect. 3, we introduce the threat model and give an overview of ACS system. In Sect. 4, we elaborate on the architecture of ACS system. After that, Sect. 5 analyzes the ACS system in detail from the resistance to active and passive attacks. Section 6 evaluates ACS system with its communication and anti-tracking performance. Finally, we conclude our work in Sect. 7.

2 Related Works

All the anonymous networks share the common goal of hiding users' identities and communication patterns from the adversary. Anonymous networks can be mainly divided into three categories: (i) Multi-hop based methods, (ii) Mix-net based methods, (iii) DC-net based methods.

2.1 Multi-Hop Based Methods

Multi-hop based methods use several relay nodes to transfer information and each relay node only knows its direct communication nodes to hide the whole transmission path. Tor, as the most popular anonymous network, provides sender anonymity through multi-hop onion routing. Considering the limitations of Tor, such as the directory server, untrusted volunteer ndoes et al., many systems propose the improvement methods based on Tor. PIR-Tor [12] is an architecture for the Tor network in which users obtain information about only a few onion routers using private information retrieval techniques(PIR), and the security of PIR-Tor depends on the security of PIR schemes. SGX-Tor [13] is a practical approach to effectively enhance the security and privacy of Tor by utilizing Intel SGX, a commodity trusted execution environment. SGX-Tor can prevent code modification and limits the information exposed to untrusted parties. Herd [14] is an anonymous network where a set of dedicated, fully interconnected cloud-based proxies yield suitably low-delay circuits, while untrusted superpeers add scalability. Herd provides caller/callee anonymity among the clients within a trust zone

and under a strong adversarial model. HORNET [15] enables high-speed end-to-end anonymous channels by leveraging next generation network architectures. HORNET is designed as a low-latency onion routing system that operates at the network layer thus enabling a wide range of applications. HORNET uses only symmetric cryptography for data forwarding yet requires no per-flow state on intermediate nodes. TresMep [16] uses node ring to relay message and each node ring randomly chooses the exit node to deliver the message. TresMep achieves a dynamic multi-hop communication path to improve tracking-resistance at the cost of communication latency.

Multi-hop based methods have the advantage in network scalability and low-latency communication, but these methods are vulnerable to traffic analysis [17,18] and malicious node infiltration [7,19,20]. Network traffic on anonymous communication has its special features that can be distinguished from the background traffic. The adversary can monitor the network traffic, recognize the anonymous traffic through its features and trace its transmission path. Moreover, it is also difficult to prevent the malicious nodes from infiltrating in the communication channel. Then, the malicious nodes can observe the communication relationship to break the anonimity.

2.2 Mix-Net Based Methods

Mix-net based methods use mix nodes to shuffle the traffic and output them in a reshuffled form. Then, the input-output relation between different traffic can be hidden, such that an adversary is not able to establish a correlation between input and output traffic. Vuvuzela [21] is a new scalable messaging system that offers strong privacy guarantees, hiding both message data and metadata. Vuvuzela is secure against adversaries that observe and tamper with all network traffic, and that control all nodes except for one server. But Vuvuzela operates in rounds, and offline users lose the ability to receive messages and all messages must traverse a single chain of relay servers. Stadium [22] and Anon-Pop [23] improve the Vuvuzela by making the routing of messages dependent on the dynamics of others. Loopix [24] provides bi-directional "third-party" sender and receiver anonymity and unobservability by leveraging cover traffic and brief message delays. Loopix allows offline users to receive messages and uses parallel mix nodes to improve the scalability of the network. Riffle [25] consists of a small set of anonymity servers and a large number of users, and guarantees anonymity as long as there exists at least one honest server. Riffle uses a new hybrid verifiable shuffle technique and private information retrieval for bandwidth and computation-efficient anonymous communication. But Riffle can not handle network churn.

Mix-net based methods can effectively resist the traffic analysis and correlation analysis. But, the computation overhead in mix nodes may result in the high latency in communication. In general, the mix nodes need to be deployed specially, so that mix-net based methods is vulnerable to the single point failure.

2.3 DC-Net Based Methods

DC-net protocol offers non-interactive anonymous communication using secure multi-party computation with information-theoretically secure anonymity, guaranteeing sender anonymity while enabling all participants to verify the final outcome [26].

Dissent [27,28] offers provable anonymity with accountability for moderate-size groups, and efficiently handles unbalanced loads where few members wish to transmit in a given round. Each DC-net run transmits the variable-length bulk data comprising one member's message, using the minimum number of bits required for anonymity. BAR [29] combines broadcasting features of DC-net with layered encryption of Mix-net. The main advantage of BAR over other broadcast systems is bandwidth configurability and it can significantly reduce the required bandwidth for a small increase in latency, without affecting anonymity. Atom [30] is an anonymity system that protects against traffic-analysis attacks and avoids the scalability bottlenecks of traditional anonymity systems. Atom consists of a distributed network of mix servers connected with a carefully structured link topology. Atom is designed for latency tolerant unidirectional anonymous communication applications with only sender anonymity in mind.

DC-Net based methods protect the users against traffic analysis attacks effectively, but sacrifice the bandwidth. The DC-net protocol lacks the flexibility in anonymous communication, which needs all participants' cooperations, and allows only one user to communicate in one protocol round.

3 Overview of ACS

3.1 Threat Model

The main goal of ACS system is to hide the users' network identities and communication patterns from the adversary. So, ACS considers adversaries with the following capabilities.

Firstly, the adversary has a global view of the network to observe all network traffic between users, entry servers and exchange servers. This adversary is able to observe the entire network infrastructure, launch network attacks or conduct indirect observations. Secondly, the adversary has the ability to corrupt the servers of ACS system. If the adversary controls the entry server, it will get the communication relationship between the users and exchange servers. If the adversary control the exchange servers, it will know the message exchange among the user' pseudonyms. But, we assume only a fraction of entry servers and exchange servers can be corrupted or be operated by the adversary. Finally, the adversary has the ability to participate in ACS system as a compromised user, who may deviate from the protocol of ACS system. But we assume that the adversary can only control a limited number of such users.

We also consider the adversary has the limited computational resources so that it cannot break the security of cryptography. In ACS system, the only leaked

information is whether an user is online or offline. This information is impossible to hide from the adversary, and hence it is the minimum level of information leakage.

3.2 The Overview

The ACS system's architecture contains of two layers, entry layer and exchange layer, as illustrated in Fig. 1. We consider a population of U users communicating through ACS, each of which can act as sender and receiver, denoted by indices s_i and r_i, where $i \in \{1, ..., U\}$. Each sender creates different shares of the message using additive secret sharing scheme [31] and sends the message shares to different entry servers randomly. The entry servers are used for grouping the received message shares and forwarding them to/from exchange servers. For anonymity, the users only provide their pseudonyms to exchange servers, and negotiate the symmetric keys with exchange servers via identity-based cryptography [32–34] to encrypt the message shares. So, all users need to register in exchange servers with pseudonyms, and exchange servers achieve the message exchange according to the pseudonyms of the communicating parties.

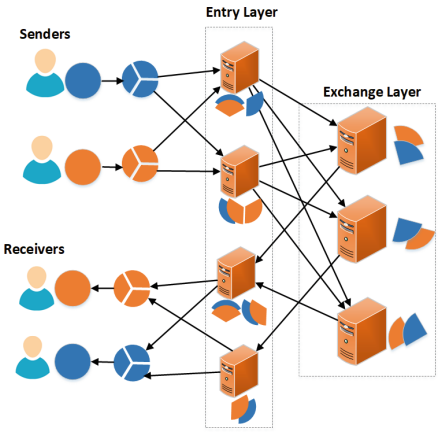


Fig. 1. The ACS system's architecture.

The exchange servers can be viewed as "information center" to exchange the message secretly and anonymously. The exchange servers also provide storage service to achieve the offline messaging. If the senders forward the message shares to exchange servers, but no receivers request for them. Exchange servers will store these message shares and wait for the requests from the receivers until the storage service of these message shares is expired. Storage service of message shares reduces the process of communication handshake between senders and receivers which makes the anti-tracking communication more flexible and efficient.

ACS contains the following three steps to implement its functionality:

- **User Registration.** At the beginning, each user u_i registers in the exchange servers with a unique pseudonym. Each exchange server takes the role of Key Generation Center (KGC) to generate the secret key for each user using a master secret key through identity-based key arrangement protocol (ID-based KAP) [32–34].
- **Anti-tracking Communication.** ACS system works in rounds. In each round of communication, all the online users need to send messages (the real messages or the cover messages) to exchange servers. The real messages contain the pseudonyms of the sender and the receiver for message exchange. The cover messages contain no useful information, but show the online status to exchange servers. All the messages will be divided into different shares, and sent to exchange servers through different entry servers. The exchange servers output the new message shares according to the content of the received message shares, and send them back to the corresponding users. After each user collects all the message shares from exchange servers, it can reconstruct the complete message.
- **Offline Messaging.** The message exchange is implemented only by the pseudonyms of users and the content of message shares. Because the exchange servers cannot receive the message shares from offline users, the messages sent to the offline users cannot be transmitted successfully. In this case, the exchange servers provide the storage service for these messages and wait the corresponding users go online. Storage service makes that the communication between the sender and receiver does not have to happen in one same round of ACS, which also improves the tracking-resistance.

4 The ACS Architecture

In this section, we will introduce the architecture of ACS system in details. For the convenience of discussion, we assume that ACS consists of three exchange servers in exchange layer: MS_1, MS_2, MS_3, and four entry servers in entry layer: ES_1, ES_2, ES_3, ES_4. We denote n as the number of users.

4.1 User Registration

At the beginning, each exchange server $MS_l, l \in \{1, 2, 3\}$, plays the role of KGC to generate the master secret key msk_l according to ID-based KAP [32–34]. Each user u_i generates a unique pseudonym UN_i of 64 bits, and sends its UN_i to each exchange server. According to the pseudonym of each user, each exchange server MS_l generates the secret key $sk_{i,l}$ for each user u_i and sends $sk_{i,l}$ to u_i. For the convenience of key agreement between users and exchange servers, each exchange server also generates its pseudonym SN_l and computes the corresponding secret key sk_{MS_l} using the master secret key msk_l. Then, each exchange server publishes its pseudonym SN_l to each user for the key agreement.

When registration is completed, each user u_i will get three secret keys: sk_1, sk_2, sk_3, and the corresponding pseudonyms of exchange servers: SN_1, SN_2,

SN_3. Each exchange server will get the list of all registered users' pseudonyms: $C_{UN} = \{UN_1, UN_2, ..., UN_n\}$. Then, users and exchange servers only use the pseudonyms to recognize each other.

4.2 Anti-tracking Communication

Anti-tracking communication is the core function of ACS. In each communication round r, the main phases of anti-tracking communication are outlined as follows.

Input preparation. In each round r, all users have to play one of the following roles: (i) the sender, which send message to a specific user, (ii) the receiver, which check the message from a specific user, and (iii) the cover, which send the cover message to exchange servers in order to circumvent the threat of traffic analysis and show the online status to ACS system.

According to the three role types, we define the message block as the form (L, u_i, u_j, M) in which L denotes the role type, u_i denotes the source user of this message, u_j denotes the target user of this message, and M denotes the context of communication. In ACS system, $L = S$ denotes the sender, $L = R$ denotes the receiver, and $L = C$ denotes the cover. We can set $M = m_c$ (m_c denotes the cover message), if no message need to be delivered, and $u_j = 0$ when the covers create the message block because the covers only send the redundant traffic.

Each sender s_i first creates three shares of its message m_i using additive secret sharing scheme. The additive secret sharing scheme works in a ring $\mathbb{Z}_N$, where a secret value $x \in \mathbb{Z}_N$ which will be shared among n parties. The n shares of x can be created by first choosing $n-1$ random numbers modulo N and the last share is computed by subtracting the $n-1$ random numbers from x and then modulo N:

$$\begin{aligned} x_1 &\leftarrow random()\ (mod\ N) \\ x_2 &\leftarrow random()\ (mod\ N) \\ &\cdots \\ x_n &\leftarrow x - x_1 - \cdots - x_{n-1}\ (mod\ N) \end{aligned} \tag{1}$$

So that, $m_i = m_{i,SM_1} + m_{i,SM_2} + m_{i,SM_3}$ according to additive secret sharing scheme, and m_{i,SM_l} denotes the share will be delivered to exchange server SM_l.

Assume the sender s_i intends to send message m_i to receiver r_j, then it creates the message share $(S, s_i, r_j, m_{i,SM_l})$, $l \in \{1, 2, 3\}$. The receiver r_j creates the message share $(R, r_j, s_i, 0)$ which denotes r_j has message request from s_i. Each cover c_k creates the message share $(C, c_k, 0, 0)$.

According to ID-based KAP, each user u_i computes the encrypted keys $k_{i,l}$ with its secret key sk_i and the pseudonym SN_l of exchange server MS_l. The exchange server MS_l can also compute the encrypted key $K_{i,l}$ with its secret key sk_{MS_l} and the pseudonym UN_i of each user u_i. Then, each user u_i uses secret key $k_{i,l}$ to encrypt the message share which will be sent to exchange server MS_l.

Anti-Tracking Communication. Users and exchange servres have no identity information about each other except their pseudonyms. So, the entry

servers relay the messages between users and exchange servers by mapping the pseudonyms to the real addresses.

Each user u_i randomly sends its message share to an entry servrer, then the entry server will get different message shares from different users. Entry servers record the corresponding relationship between users and message shares. Likewise, exchange servers record the corresponding relationship between entry servers and message shares. In this way, entry servers break the direct links between users and exchange servers and exchange servers only use the users' pseudonyms to complete the message exchange.

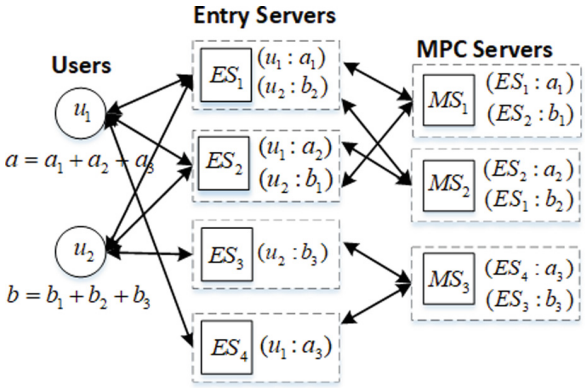


Fig. 2. The communication between users and exchange servers via entry servers.

In Fig. 2, a simple example of communication between users and exchange servers via entry servers is presented. The entry servers forward the message shares to exchange servers, and the exchange servers can recognize that each message share belongs to which entry server. After the message exchange is completed by the exchange servers, each exchange server computes the output message shares which contain the result of messaging, and sends them back to the corresponding entry servers. Then, the entry servers send these message shares to the relevant users according to the pseudonym in each message share.

Message Exchange. $MS_l (l \in \{1, 2, 3\})$ will receive one message share from each online user. To check the consistency of the message shares received by each exchange server, the exchange server first sorts the message shares according to the users' pseudonyms using the oblivious quicksort algorithm [35]. Then, each exchange server computes a hash value of the pseudonyms in the order they appear in the above sequence, as $H_{MS_l} = H(UN_1||UN_2||...||UN_n)$. Each exchange server MS_l compares its H_{MS_l} with the hash value computed by other exchange servers. If the hash values are equal, all the exchange servers receive the right message shares. Otherwise, the malicious operations are taken place in either entry servers or exchange servers.

The message shares of the *sender*, *receiver* and *cover* are respectively denoted as: $b_s = (S, s_i, r_j, m)$, $b_r = (R, r_j, s_i, 0)$, and $b_c = (C, c_k, 0, 0)$.

According to the role type and the users' pseudonyms, the exchange servers implement the message exchange process among different message shares. We conclude five cases in the message exchange process as follows:

(1) $b_s = (S, u_i, u_j, m)$ and $b_r = (R, u_j, u_i, m_{req})$. In this case, b_s and b_r have the same communicating parties, which denote user u_i and u_j wish to exchange messages. The exchange server computes the output message shares, $b'_s = (S, u_i, u_j, m_{req})$ and $b'_r = (R, u_j, u_i, m)$, and sends b'_s and b'_r to user u_i and u_j respectively. The fourth component m_{req} of b'_s denotes the request information of u_j, and the fourth component m of b'_r is the message which will be sent to u_j.

(2) $b_s = (S, u_i, u_j, m)$ and $b_c = (C, u_j, 0, 0)$. In this case, user u_i sends message to u_j, but user u_j does not request messages from u_i. The exchange server computes the output message shares, $b'_s = (S, u_i, j_j, info_s)$ and $b'_c = (C, u_j, u_i, m)$, and sends b'_s and b'_c to user u_i and u_j respectively. b'_s changes its fourth component with $info_s$ from b_s to notice u_i the completion of message exchange, b'_c changes its third and fourth components with u_i and m from b_c respectively to notice u_j that u_i has sent the message m.

(3) $b_r = (R, u_i, u_j, m_{req})$ and $b_c = (C, u_j, 0, 0)$. In this case, user u_i has a message request from user u_j, then the exchange server computes the output message shares, $b'_r = (R, u_i, u_j, info_r)$ and $b'_c = (C, u_j, u_i, m_{req})$, and sends b'_r and b'_c to user u_i and u_j respectively. b'_r changes its fourth component with $info_r$ from b_r to notice u_i that its request information has been sent to u_j, b'_c changes its third and fourth components with u_i and m_{req} from b_c respectively to notice u_j that u_i has a request message m_{req}.

(4) $b_s = (S, u_i, u_j, m)$, but no message shares come from u_j. In this case, we think user u_j is offline. Then, the exchange server stores the message share from u_i until b_s is expired. Also, the exchange server computes the output message share $b'_s = (S, u_i, u_j, info_{off})$ and sends b'_s to user u_i to notice the offline status of user u_j.

(5) $b_r = (R, u_i, u_j, m_{req})$, but no message shares come from u_j. In this case, the exchange server stores the message share from u_i, computes the output message share $b'_r = (R, u_i, u_j, info_{off})$ and sends b'_r to user u_i to notice the offline status of user u_j.

In fact, the above process of message exchange is implemented among the received message shares and the stored message shares in each exchange server. The detailed message exchange process will be discussed in the following section.

4.3 Message Storage

In each round r, all online users need to send messages to ACS system. In other words, if the exchange servers do not receive the message share from user u_k, we believe user u_k is offline. If the valid message shares, $b_s = (S, s_i, r_j, m_{i,SM_l})$ and

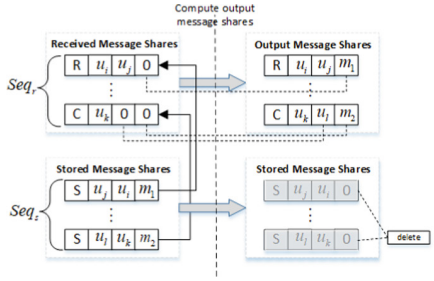
$b_r = (R, r_j, s_i, 0)$, do not match the corresponding users, the exchange servers will store them for the future message exchange.

The message exchange between the received message shares and the stored message shares in each exchange server is the key problem for offline messaging. Assume that $Seq_r = (a_1, a_2, ..., a_n)$ and $Seq_s = (b_1, b_2, ..., b_m)$ denote the sequences of received message shares and stored message shares respectively. *Source(x)* and *Target(x)* denote the source user and target user of message share x respectively. We discuss the offline messaging from the following cases:

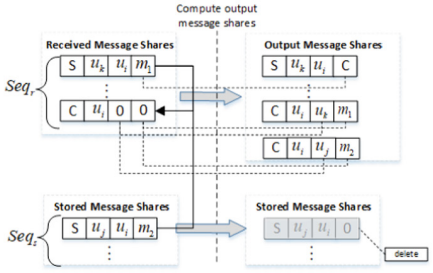
(1) Exist $a_i \in Seq_r(1 \leq i \leq n)$, $b_j \in Seq_s(1 \leq j \leq m)$, $Source(a_i) = Target(b_j)$. But for other $a_k \in Seq_r(1 \leq k \leq n, k \neq i)$, $Source(a_i) \neq Target(a_k)$. In this case, the message exchange only exists between the received message shares and stored message shares. As illustrated in Fig 3a, the exchange server extracts the valid information from the stored message shares, and computes the output message shares according to the communicating message shares.

(2) Exist $a_i, a_k \in Seq_r(1 \leq i, k \leq n, i \neq k)$, $b_j \in Seq_s(1 \leq j \leq m)$, $Source(a_i) = Target(a_k)$ and $Source(a_i) = Target(b_j)$. In this case, the received message share and stored message share have the same user for messaging. As illustrated in Fig 3b, the exchange message outputs two message shares, one is to carry the valid information of the received message share, the other is to carry the valid information of the stored message share.

In each messaging round, the exchange server deletes the stored message shares when their valid information has been transmitted or they are expired.



(a) A case of message exchange which only exists between received message shares and stored message shares.



(b) A case of message exchange in which both the received message share and stored message share have the same user for messaging.

Fig. 3. The two cases of message exchange between the received message shares and the stored message shares.

5 System Analysis

In this section, we present the analysis of ACS' security and argue its resistance to active attack and passive attack.

5.1 Resistance to Active Attack

Security of Entry Server. In the architecture of ACS, users and exchange servers have no valid information with each other except their pseudonyms. They recognize and communicate with each other only through their pseudonyms. Then, entry servers take the role of "middleman" in the communication between users and exchange servers. The advantage of entry servers is concluded as follows:

- The identities of users and exchange servers are protected by entry servers.
- The message can be mixed and shuffled by entry servers which foil the network monitors from learning about the correspondences between the users and exchange servers.
- The entry servers facilitate the synchronization and consistency of different exchange servers.

All the communication between users and exchange servers rely on entry servers, once the entry servers are corrupted by the adversary, the performance of trakcing-resistance of ACS would be threatened. So, the security of entry servers has to be taken into account. To this end, we guarantee that the number of entry servers is n_e, the number of exchange servers is n_x, and $n_e >> n_x$. The message from each user should be divided into n_m shares, and each share can only be delivered by one entry server. In this way, we can reduce the risk of collusion of corrupted entry servers.

Assume the adversary can not corrupt all the entry servers and the number of corrupted entry servers is $n_c (1 \leq n_c \leq n_e)$. Each user randomly chooses one entry server for one share, then the probability of that all chosen entry servers are corrupted is P_c shown in Equ. 2.

$$P_c = \prod_{i=1}^{n_c} \frac{n_c - (i-1)}{n_e - (i-1)} \tag{2}$$

Only $n_c \approx n_e$, then $P_c \approx 1$. In other words, if most of the entry servers are corrupted, the adversary can control the communication between the users and exchange servers in high probability. But in practical application, it is difficult for the adversary to control all the entry servers.

Even some of the entry servers may be corrupted, the corrupted entry servers can not collect all the shares from one user. Then, every malicious operation, such as tamper or discard the message shares, will be detected by exchange servers because at least one honest entry server will forward the correct information to exchange servers.

Security of Exchange Server. We consider the corrupted exchange servers which provide the wrong outputs to destroy the reconstruction of messages. Then, at least one honest exchange server can guarantee the security of ACS. Before the message exchange, each exchange server will check the consistency of all its received message shares with other exchange servers. The corrupted exchange servers which provide the wrong verification information will be detected by other exchange servers.

Each user gets different sets of ID-based KAP from different exchange servers, and encrypts the message share sent to the exchange server MS_i with the agreed upon key which is generated by MS_i. In this way, the exchange server cannot decrypt the message shares sent to other exchange servers. So, ACS guarantees the security of message shares sent to different exchange servers and deters the collusion of corrupted exchange servers.

5.2 Resistance to Passive Attack

ACS system relies on the entry servers to provide the unlinkability between the users and exchange servers. Every online user sends the messages to exchange servers, and receives the corresponding output messages from the exchange servers in each messaging round. No matter whether the users have the messaging tasks or not, they have the same communication behaviors which provide high resistance to traffic analysis and correlation attack.

Each user has no information about the exchange servers, and communicate with them only via their pseudonyms. Only entry servers know the communication relationship between the users and exchange servers. But, entry servers have no information about the message exchange process.

ACS system relies on the exchange servers to break the communication relationship between different users. All the message exchange process is completed by the exchange servers. The exchange servers provide the following advantages: (i) No information about message exchange will be leaked without the control of exchange servers. Only the exchange servers know the communication relationship among different pseudonyms of users. (ii) The message exchange only relies on users' pseudonyms, without the collusion of entry servers, no adversary can trace the communication relationship of different users. (iii) Each exchange server also provides the message storage service for offline messaging, so the message exchange process may not happen in one messaging round of ACS system. It is nearly impossible for the adversary to trace the communication of different users which happens in different messaging rounds.

The messages transmitted in ACS system may be the real or the cover messages. All the messages will be created into different shares and encapsulated into the same form (L, u_i, u_j, M) which has been mentioned above. All message shares are padded to the same length and end-to-end encrypted to make sure the adversary can not distinguish the real message from the cover message, and also learn no information from the encrypted messages.

6 Experiments and Evaluation

We implement a prototype of the ACS system which contains 5 exchange servers which are deployed on the computer with a 12-core 4 GHz CPU and 64 GB RAM, and 10 entry servers which are deployed on Cloud platforms with a 2-core 4 GHz CPU and 48 GB RAM. We run a simulation program on a computer with a 8-core 3 GHz CPU and 64 GB RAM to simulate the independent users to communicate with the ACS system.

We evaluate the ACS prototype system from its system performance and anti-tracking ability.

6.1 Performance Evaluation

We first evaluate the latency overhead of ACS system in consideration of the various number of users and the different message size. In this experiment, all users send the messages to a randomly chosen target in each round, and the message size is set with $1KB, 2KB, 4KB$ and $8KB$ respectively. After the prototype system completes the communication, we measure the running time to evaluate the latency overhead. In the following experimental results, we use $|M|$ to indicate the message size.

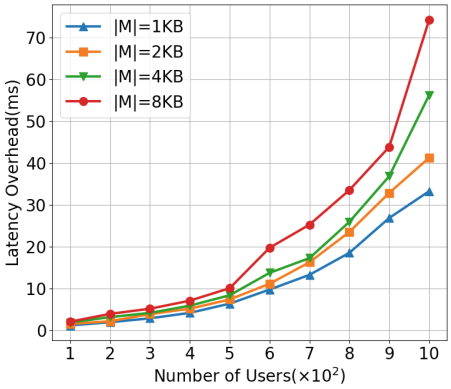
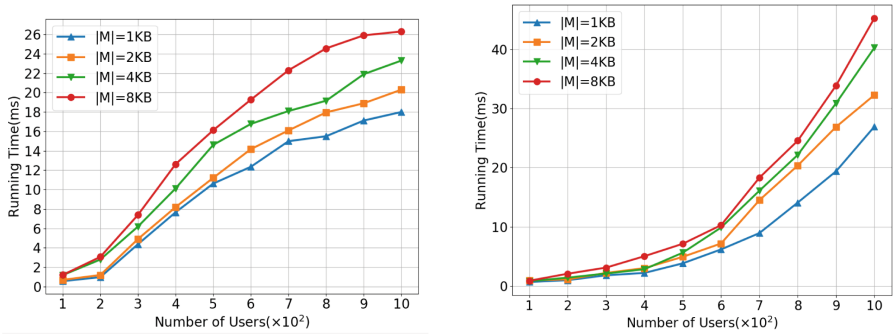


Fig. 4. Latency overhead of ACS system where 100 to 1000 users simultaneously send message with different size.

As shown in Fig. 4, with the increasing of the number of users and the message size, we can see that the latency overhead of the prototype system grows exponentially. In the current system configuration, when the number of users is less than 500, the latency overhead of all the four cases is in a reasonable range.

But, when the number of users exceeds 500, the latency overhead grows sharply. There are two aspects which may affect the communication efficiency of ACS system: (i) the performance of message forwarding of entry servers, and (ii)

the performance of message exchange in exchange servers. Next, we will evaluate the performance of the entry servres and exchange servers respectively to find out the bottleneck of ACS system in communication efficiency.



(a) The running time of message exchange of all exchange servers in different message size.l

(b) The running time of Entry servers in different message size.

Fig. 5. The running time of entry servers and exchange servers in different message size.

To evaluate the performance of exchange servers, we directly send the message shares to exchange servers and measure the running time when exchange servers complete the message exchange. As shown in Fig. 5a, all the curves grow sharply when the number of users exceeds 500. The message exchange in each exchange server uses the oblivious Quicksort algorithm [35] which needs $O(log(n))$ steps to sort n inputs when executed in parallel [36]. So, the complexity of message exchange computation is in logarithm relation with the number of inputs. The exchange servers need more computation cost to match the communicating users and computes the output messages when the number of users is increased.

To evaluate the performance of entry servers, we only implement the function of entry servers to relay the message shares. The exchange servers do not take any operations and directly send back the received messages to entry servers as the outputs. Then, we measure the running time of entry servers when they complete the transmission of all the upstream and downstream messages. As shown in Fig 5b, with the increasing of the number of users, the running time of entry servers grows exponentially. Compared with the experimental results of latency overhead evaluation, the performance of entry servers in message forwarding has a direct effect on the communication efficiency of ACS system. In our prototype system, we only set five entry servers to relay the messages between the users and exchange servers. With the increasing of users, entry servers need to process more transmission requests from users. The entry servers also need extra time to mix, label the message shares and correlate the pseudonyms with their real network addresses.

6.2 Anti-tracking Evaluation

To effectively evaluate the anti-tracking performance of ACS, we introduce a measuring method of tracking-resistance based on information entropy [37,38]. Let X be a discrete random variable over the finite set $\mathbb{F}$ with probability mass function $p(x) = P(X = x)$. The Shannon entropy H(X) of a discrete random variable X is defined as Equ. 3.

$$H(X) = -\sum_{x \in X} p(x) log(p(x)) \tag{3}$$

In ACS, the entry server protects the privacy of users' network identities, the exchange server protects the privacy of the message exchange information among users' pseudonyms. An adversary has to steal the above two kinds of information to trace the communication relationship. Assume that the number of entry servers is n_e and n'_e entry servers are corrupted by the adversary. The number of exchange servers is n_x, and n'_x exchange servers are corrupted. The corrupted entry server can analyze the message shares and distinguish the sender and target exchange server of each message share, but cannot decrypt the message share encrypted by the secret key of each exchange server. The corrupted exchange server knows the pseudonyms in communication. So, the path between two communicating users contains three hops: $v_e \rightarrow v_x \rightarrow v_e$, in which v_e denotes the entry server and v_x denotes the exchange server. The traceable probability of ACS, denoted by P_{trace}, is defined by Eq. 4.

$$P_{trace} = (\frac{n'_e}{n_e})^2 \frac{n'_x}{n_x} \tag{4}$$

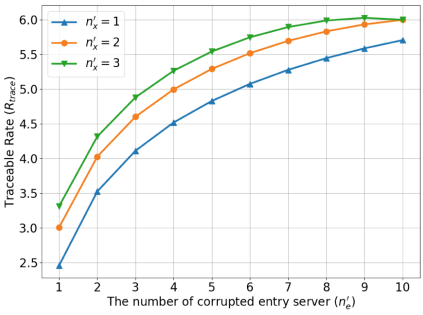
We consider that the messaging behavior of each user is independent in probability, and each user randomly chooses the entry servers to forward the message shares. Then, the distribution of traced users can regard as Binomial Distribution. For n communicating users, k of them can be traced by the adversary, then the probability of traced users is defined by Eq. 5.

$$P\{X = k\} = \binom{n}{k} (P_{trace})^k (1 - P_{trace})^{n-k} \tag{5}$$

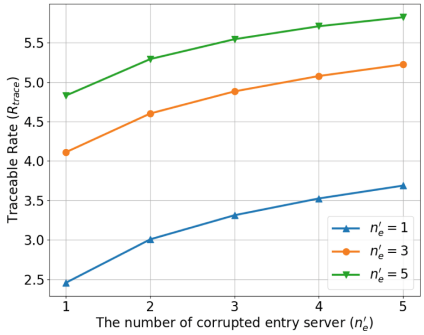
The traceable rate R_{trrace} indicates the probability of tracing the communicating users via the corrupted servers. We use the entropy of traced users' distribution to describe R_{trace}. So, R_{trace} can be calculated by Eq. 6.

$$R_{trace} = -\sum_{k=1}^{n} (P\{X = k\}) log(P\{X = k\}) \tag{6}$$

We simulate 1000 users to communicate with each other via ACS prototype system. Through the increase of n'_e and n'_x, we evaluate the tracking-resistance of ACS according to its traceable rate R_{trace}. In Fig. 6a, we increase n'_e from 1 to 10, and compare R_{trace} under different n'_x. In Fig. 6b, we increase n'_x from 1 to 5, and compare R_{trace} under different n'_e. From the experimental results, R_{trace} increases in logarithm relation with both of n'_e and n'_x. The adversary can reach an efficient traceable rate R_{trace} only when both of the entry servers and exchange servers are compromised in high probability. So, the anti-tracking ability lies in the double protection of entry server and exchange server. On the contrary, it's not sufficient for the adversary to trace the communicating users if they only compromise one kind of servers.



(a) Traceable rate w.r.t. corrupted entry servers.



(b) Traceable rate w.r.t. corrupted exchange servers.

Fig. 6. The tracking-resistance evaluation with increasing number of corrupted entry servers and corrupted exchange servers.

7 Conclusion

In this paper, we present ACS system to achieve the anti-tracking messaging. In the design of ACS system, ACS contains two layers: entry layer and exchange layer, which separates the two process of message forward and message exchange. The entry servers are only used to relay messages between the users and exchange servers. The exchange servers are only used to exchange the message shares. To improve the anti-tracking performance, all the users use pseudonyms to label their messages, and exchange servers have no information about the users' real identities. Even the entry servers know the identities of the users, but they have no information about the message exchange process. So, the adversary has to corrupt both the entry servers and exchange servers to trace the communicating users. In addition, message storage is also provided to achieve offline messaging.

We evaluate the prototype system of ACS of its communication latency and anti-tracking performance. From the experimental results, ACS has a acceptable communication latency and a strong anti-tracking ability. Only when both of the entry servers and exchange servers are corrupted, the adversary can trace the communicating users effectively. But in practical application, it is difficult for the adversary to corrupt both of the entry server and exchange server.

Acknowledgment. The authors would like to thank the anonymous reviewers for their insightful comments and suggestions on this paper. This work was supported in part by the National Key Research and Development Program of China under Grant No.2019YFB1005205.

References

1. Niaki, A.A.: Iclab: a global, longitudinal internet censorship measurement platform. In: 2020 IEEE Symposium on Security and Privacy (SP), pp. 135–151. IEEE (2020)
2. Raman,R.S.: Measuring the deployment of network censorship filters at global scale. In: Network and Distributed Systems Security (NDSS) Symposium (2020)
3. Yadav, T.K., Sinha, A., Gosain, D., Sharma, P.K., Chakravarty, S.: Analyzing web censorship mechanisms in india Where the light gets. In: Proceedings of the Internet Measurement Conference, pp. 252–264 (2018)
4. Pearce, P., Ensafi, R., Li, F., Feamster, N., Paxson, V.: Toward continual measurement of global network-level censorship. IEEE Security Privacy **16**(1), 24–33 (2018)
5. Tian, C., Zhang, Y., Yin, T.: Topology self-optimization for anti-tracking network via nodes distributed computing. In: Gao, H., Wang, X. (eds.) CollaborateCom 2021. LNICST, vol. 406, pp. 405–419. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92635-9_24
6. Johnson, A., Wacek, C., Jansen, R., Sherr, M., Syverson, P:. Users get routed: Traffic correlation on tor by realistic adversaries. In: Proceedings of the 2013 ACM SIGSAC conference on Computer and communications security, pp. 337–348 (2013)
7. Sun, Y.: {RAPTOR}: Routing attacks on privacy in tor. In: 24th {USENIX} Security Symposium ({USENIX} Security 15), pp. 271–286 (2015)
8. Tian, C., Zhang, Y., Yin, T.: A feature-flux traffic camouflage method based on twin gaussian process. In: 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 959–966. IEEE (2021)
9. Danezis, G., Dingledine, R., Mathewson, N.: Mixminion: Design of a type iii anonymous remailer protocol. In 2003 Symposium on Security and Privacy, 2003, pp. 2–15. IEEE (2003)
10. Borges, F., Buchmann, J., Mühlhäuser, M.:.Introducing asymmetric dc-nets. In: 2014 IEEE Conference on Communications and Network Security, pp. 508–509, IEEE (2014)
11. Golle, P., Juels, A.: Dining cryptographers revisited. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 456–473. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_27
12. Mittal, P., Olumofin, F.G., Troncoso, C., Borisov, N., Goldberg, I.: Pir-tor: Scalable anonymous communication using private information retrieval. In: USENIX Security Symposium, pp. 31–31 (2011)

13. Kim, S., Han, J., Ha, J., Kim, T., Han, D.: Sgx-tor: a secure and practical tor anonymity network with sgx enclaves. IEEE/ACM Trans. Netw. **26**(5), 2174–2187 (2018)
14. Blond, S.L., Choffnes, D., Caldwell, W., Druschel, P., Merritt, N.: Herd: A scalable, traffic analysis resistant anonymity network for voip systems. In: Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, pp. 639–652 (2015)
15. Chen, C., Asoni, D.E., Barrera, D., Danezis, G., Perrig, A.: Hornet: High-speed onion routing at the network layer. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 1441–1454 (2015)
16. Tian, C., Zhang, Y., Yin, T., Tuo, Y., Ge, R.: Achieving dynamic communication path for anti-tracking network. In: 2019 IEEE Global Communications Conference (GLOBECOM), pp. 1–6. IEEE (2019)
17. Montieri, A., Ciuonzo, D., Aceto, G., Pescapé, A.: Anonymity services tor, i2p, jondonym: classifying in the dark (web). IEEE Trans. Dependable Secure Comput. **17**(3), 662–675 (2018)
18. Kwon, A., AlSabah, M., Lazar, D., Dacier, M., Devadas, S.: Circuit fingerprinting attacks: Passive deanonymization of tor hidden services. In: 24th {USENIX} Security Symposium ({USENIX} Security 15), pp. 287–302 (2015)
19. Evans, N.S., Dingledine, R., Grothoff, C.: A practical congestion attack on tor using long paths. In: USENIX Security Symposium, pp. 33–50 (2009)
20. 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)
21. Van Den Hooff, J., Lazar, D., Zaharia, M., Zeldovich, N.: Vuvuzela: Scalable private messaging resistant to traffic analysis. In: Proceedings of the 25th Symposium on Operating Systems Principles, pp. 137–152 (2015)
22. Tyagi, N., Gilad, Y., Leung, D., Zaharia, M., Zeldovich, N.: Stadium: A distributed metadata-private messaging system. In: Proceedings of the 26th Symposium on Operating Systems Principles, pp. 423–440 (2017)
23. Gelernter, N., Herzberg, A., Leibowitz, H.: Two cents for strong anonymity: the anonymous post-office protocol. In: Capkun, S., Chow, S.S.M. (eds.) CANS 2017. LNCS, vol. 11261, pp. 390–412. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02641-7_18
24. Piotrowska, A.M., Hayes, J., Elahi, T., Meiser, S., Danezis, G.: The loopix anonymity system. In: 26th {USENIX} Security Symposium ({USENIX} Security 17), pp. 1199–1216 (2017)
25. Kwon, A., Lazar, D., Devadas, S., Ford, B.: Riffle: an efficient communication system with strong anonymity. Proc. Privacy Enhanc. Technol. **2016**(2), 115–134 (2016)
26. Shirazi, F., Simeonovski, M., Asghar, M.R., Backes, M., Diaz, C.: A survey on routing in anonymous communication protocols. ACM Comput. Surv. (CSUR), **51**(3), 1–39 (2018)
27. Corrigan-Gibbs, H., Ford, B.: Dissent: accountable anonymous group messaging. In: Proceedings of the 17th ACM Conference on Computer and Communications Security, pp. 340–350 (2010)
28. Wolinsky, D.I., Corrigan-Gibbs, H., Ford, B., Johnson, A.: Dissent in numbers: Making strong anonymity scale. In: 10th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 12), pp. 179–182 (2012)

29. Kotzanikolaou, P., Chatzisofroniou, G., Burmester, M.: Broadcast anonymous routing (bar): scalable real-time anonymous communication. Int. J. Inf. Secur. **16**(3), 313–326 (2017)
30. Kwon, A., Corrigan-Gibbs, H., Devadas, S., Ford, B.: Atom: Horizontally scaling strong anonymity. In: Proceedings of the 26th Symposium on Operating Systems Principles, pp. 406–422 (2017)
31. Gelernter, N., Herzberg, A., Leibowitz, H.: Two cents for strong anonymity: the anonymous post-office protocol. In: Capkun, S., Chow, S.S.M. (eds.) CANS 2017. LNCS, vol. 11261, pp. 390–412. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02641-7_18
32. Paterson., Srinivasan, S.: On the relations between non-interactive key distribution, identity-based encryption and trapdoor discrete log groups. Designs, Codes Cryptograph. **52**(2), 219–241 (2009)
33. Chen, L., Cheng, Z., Smart, N.P.: Identity-based key agreement protocols from pairings. Int. J. Inform. Security **6**(4), 213–241 (2007)
34. Wu, L., Zhang, Y., Raymond Choo, K.-K., He, D.: Efficient and secure identity-based encryption scheme with equality test in cloud computing. Future Gen. Comput. Syst. **73**, 22–31 (2017)
35. Bogdanov, D., Laur, S., Talviste, R.: A practical analysis of oblivious sorting algorithms for secure multi-party computation. In: Bernsmed, K., Fischer-Hübner, S. (eds.) NordSec 2014. LNCS, vol. 8788, pp. 59–74. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11599-3_4
36. Alexopoulos, N., Kiayias, A., Talviste, R., Zacharias, T.: Mcmix: Anonymous messaging via secure multiparty computation. In: 26th {USENIX} Security Symposium ({USENIX} Security 17), pp. 1217–1234 (2017)
37. Serjantov, A., Danezis, G.: Towards an information theoretic metric for anonymity. In: Dingledine, R., Syverson, P. (eds.) PET 2002. LNCS, vol. 2482, pp. 41–53. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36467-6_4
38. Díaz, C., Seys, S., Claessens, J., Preneel, B.: Towards measuring anonymity. In: Dingledine, R., Syverson, P. (eds.) PET 2002. LNCS, vol. 2482, pp. 54–68. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36467-6_5