Computer Networks 74 (2014) 13–33

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

# PACOM: Parasitic anonymous communication in the BitTorrent network

Jianming Lv [a,*], Tieying Zhang [b], Zhenhua Li [c], Xueqi Cheng [b]

[a] *Department of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China*
[b] *Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China*
[c] *School of Software and TNLIST, Tsinghua University, Beijing 100084, China*

A R T I C L E I N F O

*Article history:*
Received 26 May 2013
Received in revised form 3 July 2014
Accepted 24 August 2014
Available online 16 September 2014

*Keywords:*
Privacy
Anonymous communication
Peer-to-Peer
Parasitic
Traffic analysis attack

A B S T R A C T

Existing anonymous communication systems mask the identities of users by adopting intermediary nodes to transform message flows. However, some recently presented traffic analysis algorithms are still able to undermine the anonymity of these systems. The traditional flow transformation strategies fail to completely eliminate the traffic correlation between adjacent communication links to prevent such attacks. To address this problem, we propose a novel *parasitic anonymous communication system*, named PACOM. Each PACOM client is parasitic in the BitTorrent network which is the most popular Peer-to-Peer file sharing network, and conceals the communication path in the request driven traffic compatible with the BitTorrent protocol. The traffic patterns of adjacent communication links can be proved to be statistically independent, which effectively resists the traffic analysis attacks. Meanwhile, the "*effective anonymity set size*" of the system can be extended enormously by mixing the PACOM clients with other millions of BitTorrent clients in the Internet. To validate the PACOM solution, we analyse the anonymity of PACOM theoretically and conduct comprehensive simulations and emulations to test the scalability and effectiveness of PACOM against various attacks.

## 1. Introduction

As people rely on the Internet more and more in their daily life, users' anonymity has become a critical issue to protect personal privacy in a lot of Internet applications such as web browsing, file sharing, e-commerce, and electronic voting.

Existing anonymous communication systems [1–11] are generally designed to hide the relationship between the information sender and receiver from adversaries. These systems adopt intermediary nodes (often called "*mixes*") to encrypt and relay the communication messages hop by hop, so as to conceal the destinations of the messages. Besides, in order to prevent the adversaries from identifying the communication path via *traffic analysis*, some additional strategies are used to transform the network flows, such as traffic padding [1,2], cover traffic adding [4,12,13], packet dropping [14], flow mixing [15–17], batching [16,18] and rescheduling [19,20]. These transformation strategies make the network flows indistinguishable from each other.

Unfortunately, some recently presented *traffic analysis attacks* against the above mentioned systems are still applicable and can be easily conducted [21–24]. These attacks utilize the common characteristic of existing systems: the traffic patterns of adjacent links in the anonymous communication path are statistically corre-

* Corresponding author. Tel.: +86 20 39380281.
*E-mail addresses:* jmlv@scut.edu.cn (J. Lv), zhangtieying@ict.ac.cn (T. Zhang), lizhenhua1983@tsinghua.edu.cn (Z. Li), cxq@ict.ac.cn (X. Cheng).

http://dx.doi.org/10.1016/j.comnet.2014.08.015
1389-1286/© 2014 Elsevier B.V. All rights reserved.

lated. For example, on a path $A \rightarrow B \rightarrow C$, the anonymous communication data is transferred from $A$ to $C$ through $B$. The sending time of the packets in $B \rightarrow C$ depends on the arrival time and volume of the packets in $A \rightarrow B$. Thus, the traffic patterns in $B \rightarrow C$ and $A \rightarrow B$ have strong timing correlation with each other. The traditional flow transformation strategies fail to completely eliminate such correlation information. In particular, the traffic analysis attacks [21–24] can identify the communication path effectively by exploiting the traffic timing pattern and building correlation between different links.

In this paper we present a novel *parasitic anonymous communication system*, named PACOM, which is immune to these traffic analysis attacks. Each PACOM client is parasitic in the BitTorrent network, the most popular Peer-to-Peer file sharing network containing more than 150 million active users [25]. The PACOM clients pretend to share and download files in the BitTorrent network and embed the anonymous communication data into the delivered file blocks. The communication path is concealed by the request driven traffic compatible with the BitTorrent protocol, which makes the traffic patterns of adjacent links statistically independent of each other. Compared with the state-of-the-art anonymous communication systems, PACOM achieves the following advantages:

(1) PACOM is immune to the traffic analysis attacks. No statistical correlation in the time domain can be built between different links in any anonymous communication path. The success rate of the traffic analysis attacks against PACOM is close to that of random guess without any prior knowledge.
(2) PACOM is designed for two different use cases: the *Private Use Case* for secret and hidden communication among a few special users, and the *Public Use Case* for a large number of online people to transfer messages anonymously like Tor [2]. In the *Private Use Case*, PACOM equipped with steganography techniques provides strong anonymity by mixing PACOM clients with other millions of BitTorrent clients in the Internet, and prevents adversaries from sensing when and where the anonymous communication happens. In this case, the *effective anonymity set size*[1] increases logarithmically over the total number of the online BitTorrent clients. In the *Public Use Case*, no steganography is taken and PACOM is a public and open system, in which the *effective anonymity set size* is related to number of online PACOM clients. Although the number of BitTorrent clients is not helpful to increase the anonymity of PACOM in this case, the file block request driven traffic pattern following the BitTorrent protocol still makes the system effective against traffic analysis attacks and efficient in anonymous communication.
(3) Aided by the efficient file blocks transferring mechanism of the BitTorrent protocol, PACOM is efficient to transfer communication messages. Different from the traditional inefficient batching methods adopted in mix networks to resist traffic analysis, each PACOM node transfers file blocks containing communication messages driven by file block requests following the BitTorrent protocol, which is designed for efficient P2P file sharing. The bandwidth of PACOM communication in the *Private Use Case* is about 21 kB/s on average, while the bandwidth in the *Public Use Case* is about 314 kB/s on average. The end-to-end latency in both cases is close to 2.5 s. Thus PACOM in the *Private Use Case* is competent to transfer messages or files of small size. Meanwhile, PACOM in the *Public Use Case* is efficient to support various kinds of file sharing, chatting, or accessing some web based services such as cloud storage and LBS.
(4) PACOM is decentralized and highly scalable. No centralized directory servers are needed. The PACOM clients collaborate with each other to perform the network bootstrapping and maintenance operations, and the per-client computation and traffic overhead is moderate.
(5) The cover traffic in PACOM is self-adaptive and localized, thus the load of the clients is reduced and the bandwidth utilization is improved. Comprehensive experiments show that the cover traffic occupies about 50% of the bandwidth of each PACOM client, when each client initiates one communication path. The ratio drops to about 10% when each one initiates four paths on average. Moreover, the cover traffic is only produced among the PACOM clients and has little side effect to the BitTorrent network.

The remainder of this paper is organized as follows. In Section 2, we introduce some preliminaries. Then, we present the design of PACOM in Section 3 and theoretically analyse the anonymity of PACOM in Section 4. Some discussion on the deployment of PACOM is presented in Section 5. The performance of PACOM is evaluated via comprehensive simulation and emulation experiments in Section 6. In Section 7 we survey the related work. Finally, we conclude this paper in Section 8.

## 2. Preliminaries

### 2.1. Mix networks and steganography

In the past decade, a lot of anonymous communication systems [1–11] are proposed to hide the relationship between information sender and receiver from malicious adversaries. As illustrated in Fig. 1(a), internal nodes (called mixes) are adopted in these systems to mix and relay encrypted messages. To further confuse adversaries, these mixes adopt some strategies to transform all incoming flows, such as batching [15,16,18], traffic padding [1,2], cover traffic adding [4,12,13], packet dropping [14], flow mixing [15–17], and rescheduling [19,20].

Specifically, the batching has become one basic and widely used technique since the mixes were firstly

[1] An information theoretic metric of anonymity defined by Serjantov et al. [26] that quantitatively measures the anonymity level of a communication system.

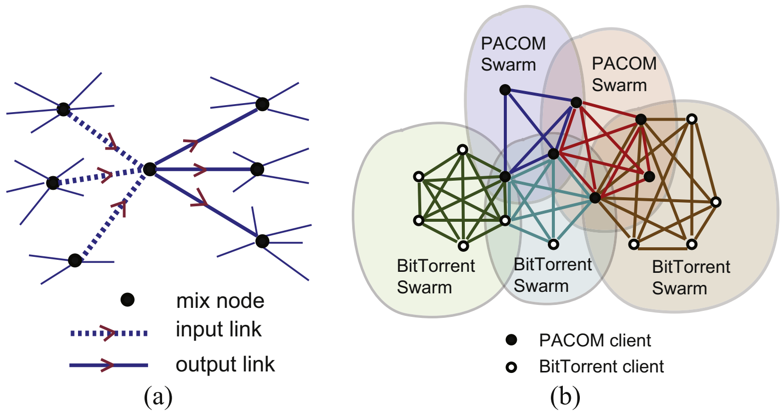

**Fig. 1.** Comparison between PACOM and traditional mix networks. (a) The traditional Mix network. (b) PACOM clients parasitic in the BitTorrent network.

proposed by Chaum and designed for anonymous email delivering in 1981 [15]. In the Chaum mix networks, mix nodes act as proxies to batch incoming mail messages from different users and send out messages in lexicographically ordered and uniformly sized items. To resist correlation analysis among links, a mix node does not relay messages until it receives enough messages from different users. That also means a relatively high latency to deliver messages can be caused by this kind of batching method. Thus the systems [15,16,27] like Chaum mixes are classified as high-latency mix networks, which are designed to deliver email or documents not urgent, and the latency can range from minutes to hours. Moreover, some recent research [21] shows that traditional batching cannot effectively eliminate the packet timing correlation between the incoming and outgoing links. Zhu et al. [21] investigate a broad range of batching methods used in mix networks, including timer based, threshold based, pool based batching, and combinations of above. The experimental results [21] show that the success rate of the flow-correlation attacks in these systems can reach 100% when enough traffic is observed.

Beyond the basic batching mechanism, the research [22] shows that the traditional flow transformation methods such as adding cover traffic, dropping packets and flows mixing/merging/splitting, do not necessarily make a network flow indistinguishable from other independent flows. Experiments indicate that all of them are weak against the active traffic analysis based on watermarking presented in [22].

In most of above systems, the traffic of the output links of a mix node is driven by the data packets arriving from input links as illustrated in Fig. 1(a). That means, if adversaries manipulate the input links by delaying, dropping, or flushing network packets, some change of the traffic pattern can be observed in the output links. The correlation of adjacent links can be leveraged by adversaries to deduce communication paths.

To solve this problem, we propose the PACOM network, which is a kind of mix network parasitic in the BitTorrent network. As illustrated in Fig. 1(b), PACOM clients perform like BitTorrent clients and join the BitTorrent network to mix with other BitTorrent clients. Each PACOM client transfers file blocks conforming to the BitTorrent protocol and embeds communication data within the blocks. The specific request-driven file block transferring mechanism of the BitTorrent protocol makes the links of each PACOM client have independent traffic pattern. In this way, beyond traditional mix networks, the PACOM network provides twofold protection of anonymity:

- Firstly, the mixing of PACOM clients and numerous BitTorrent clients makes it hard for adversaries to determine the boundary of the communication system. Thus it can increase the anonymity of the system enormously.
- Secondly, even if PACOM clients are distinguished from other BitTorrent clients, the system is still effective against the powerful traffic analysis attacks, due to the independent traffic pattern of the links.

Furthermore, in order to make the traffic of PACOM clients exactly conform to the BitTorrent protocol, we develop the PACOM client based on the BitTorrent module [50] to produce real BitTorrent traffic as illustrated in Fig. 2. Communication data is embedded into transferred file blocks. Meanwhile, steganography techniques are adopted to hide data into transferring BitTorrent messages in the following two cases:

- Hide some boostrapping information in the BitTorrent control messages by manipulating the order to download file blocks. This method is suitable for hiding information of small size. The detail is offered in the following Section 3.3 and Appendix A.

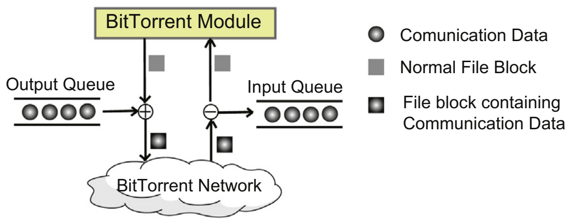

**Fig. 2.** Architecture of a PACOM client.

- Hide communication data into transferred file blocks by steganography methods as mentioned in Section 3.4.

### 2.2. BitTorrent network overview

BitTorrent [41] is one of the most popular Peer-to-Peer file sharing protocol involving more than 150 million active users. Each shared file in BitTorrent is identified uniquely by a 20 bytes' SHA-1 hash code usually named as *info_hash*. The file shared in the network is divided into pieces of equal size (between 256 kB and 4 MB normally), and each piece is further divided into blocks with a size of about 16 kB (the value varies in different implementation versions of BitTorrent).

According to the protocol specification [42], each BitTorrent client implements a distributed hash table (DHT) based on the Kadmelia protocol [43] to support distributed file searching and publishing in the network. Two main API functions implemented in the DHT are as follows:

(1) *get_peers*(*info_hash*): Each client can invoke this function to search the contact information of the clients downloading or seeding the file identified by *info_hash*.
(2) *announce_peer*(*info_hash*): A client can invoke this function to announce that it is downloading or seeding the file identified by *info_hash*. After the function is invoked, the client can be located by other ones through the *get_peers* function.

When a client plans to download a file, it builds connections and exchanges file blocks with other ones downloading or seeding the file. The overlay network formed by the clients downloading or seeding the same file is called a *swarm*. According to the observation by Guo et al. [44], each BitTorrent client joins 7.51 swarms on average, which makes most swarms connected together to form the BitTorrent network with large connected components.

## 3. PACOM

### 3.1. Design goal and assumptions

Like most anonymous communication systems, PACOM is designed to prevent adversaries from linking communication partners. Rather than the *traffic confirmation attacks* to confirm a suspicion that Alice is talking to Bob, we aim to prevent the more powerful *traffic analysis attacks*, where adversaries try to build the traffic correlation of different links in a communication path hop by hop and infer the communication partners. PACOM is designed for the following two different use cases:

- *Private Use Case:* PACOM is used by some organizations such as intelligence agencies to perform secret and private communication among their members. The PACOM client software is not public and hard to be achieved by adversaries. PACOM provides strong anonymity by mixing PACOM clients with other millions of BitTorrent clients. PACOM users can send short messages or files efficiently, while transferring large files needs a relatively long time.
- *Public Use Case:* The PACOM client is public for use like Tor [2]. Any pair of users can use PACOM to transfer messages or large files between each other anonymously and efficiently.

On the other hand, we assume that the adversaries can observe, generate, modify, delay or drop some fraction of the network traffic. Based on this primary assumption, we define two levels of threat models according to different use cases of PACOM:

- *Threat Model I:* The adversaries can compromise some fraction of BitTorrent clients and join the BitTorrent network, but cannot compromise PACOM clients. This threat model is corresponding to the *Private Use Case* of PACOM for private and secret communication. The secret communication key embedded in the PACOM client is not public and is not easy to compromise.
- *Threat Model II:* Behaving more aggressively, the adversaries can compromise some fraction of BitTorrent clients as well as PACOM clients. This model can be applied to PACOM for the *Public Use Case*, where every one can use a PACOM client to initiate anonymous communication.

We will analyse in Section 4 the anonymity of PACOM against different attacks under both threat models mentioned above.

### 3.2. PACOM network model

As shown in Fig. 2, the PACOM client is constructed based on the BitTorrent module [50], which makes the traffic exactly compatible with the BitTorrent protocol. PACOM clients join the BitTorrent network and perform like normal BitTorrent clients. As illustrated in Fig. 1(b), PACOM clients join some swarms created by BitTorrent clients (named 'BitTorrent swarm' in the paper) to download valid files. Meanwhile, PACOM clients create swarms (named 'PACOM swarm' in the paper) among themselves to transfer some files embedding communication data. Equipped with some carefully designed constraints, all PACOM swarms can be linked into the connected PACOM network which is mixed with the BitTorrent network.

In the PACOM network, each client schedules the packets conforming to the request driven pattern of the BitTorrent protocol as illustrated in Fig. 3. The communication data segments mixed with *dummy* data are embedded in the transferred file blocks. The traffic of each link is independent of the arrival time and the volume of the embedded communication data, and is only triggered by the file block requests following the BitTorrent protocol. For example, in the communication path $P_1 \rightarrow P_i \rightarrow P_4$ of Fig. 3, the traffic in $P_1 \rightarrow P_i$ is triggered by the block requests from $P_i$, while the traffic in $P_i \rightarrow P_4$ is triggered by the requests from $P_4$. Since each client initiates block

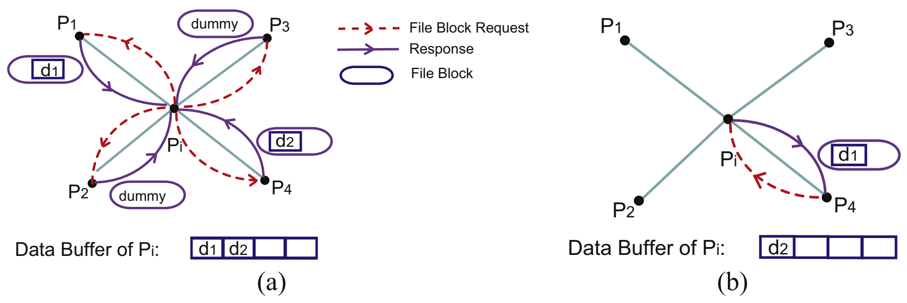

**Fig. 3.** The packet scheduling of the PACOM client $P_i$. $P_1 \rightarrow P_i \rightarrow P_4$ is a concealed communication path. $P_i$ forwards the data segment $d_1$ sent from $P_1$ to $P_4$ in the path. (a) Compatible with the BitTorrent protocol, $P_i$ requests its neighbors for its missing file blocks. Each neighbor then responds with a file block embedding an encrypted communication data segment sent to $P_i$. If a neighbor has no data segment ready to send out, it embeds the dummy data in the file block as response. $P_i$ records the data segments received from its neighbors into the local buffer. (b) $P_i$ does not forward $d_1$ to $P_4$ until it receives a file block request from $P_4$. $P_i$ then encrypts $d_1$ and embeds it in the file block as response.

**Table 1**
Notations in Algorithm 1.

| | Description |
|---|---|
| $S(P_i)$ | The collection of swarms that $P_i$ joins |
| $B(P_i)$ | The collection of file blocks owned by $P_i$ |
| $Q(P_i, P_j)$ | The output queue of $P_i$ storing the data for $P_j$ |
| $R(P_i, P_j)$ | The set of the unsolved requests from $P_i$ to $P_j$ |
| $S_k$ | A swarm |
| $F(S_k)$ | The file shared in the swarm $S_k$ |
| $C(S_k)$ | The collection of clients joining the swarm $S_k$ |
| $d_k$ | A communication data segment |
| $b_k$ | A file block |

requests independently according to the BitTorrent protocol, the adjacent links $P_1 \rightarrow P_i$ and $P_i \rightarrow P_4$ have independent traffic patterns. No correlation can be built to infer the existence of the concealed communication path $P_1 \rightarrow P_i \rightarrow P_4$ through traffic analysis.

The detailed packet scheduling procedure on each PACOM client is illustrated as Algorithm 1 and the notations used in the algorithm are listed in Table 1. The procedure can be divided into three parts: “sending request”, “receiving response”, and “receiving request and responding”. Compatible with the BitTorrent protocol, each PACOM client requests other clients in the same swarm for its missing file blocks. The requested blocks are selected in a randomized manner. To avoid overwhelming a client with too many requests, the unsolved requests sent to any client is limited within a constant $\theta$, which is set as 10 according to the BitTorrent Specification [45]. While the client receives a response with a file block from another PACOM client, it decodes the communication data segment within and places the segment into the corresponding output queue ready for forwarding. When the client receives a block request from another PACOM client, it tries to get a data segment from the queue to compose the response packet. If the queue is empty at that time, the client constructs the response with dummy data. In this way, the traffic of each client is driven by the file block requests, rather than the embedded communication.

**Algorithm 1.** The packet Scheduling of a PACOM client.

```
Input: $P_i$: A PACOM client.
Output: The packet scheduling of $P_i$.
Method:
 1 for $\forall S_k \in S(P_i)$ and $\forall P_j \in C(S_k)$ $(i \neq j)$ do
 2   //Sending request
 3   if $B(P_j) - B(P_i) \neq \emptyset$ and $|R(P_i, P_j)| < \theta$ then
 4     $b \leftarrow$ a block randomly selected from $B(P_j) - B(P_i)$
 5     $P_i$ sends to $P_j$ a request for $b$
 6     $R(P_i, P_j) \leftarrow R(P_i, P_j) \cup \{b\}$
 7   //Receiving response
 8   if $P_i$ receives a block $b_k$ from $P_j$ then
 9     $R(P_i, P_j) \leftarrow R(P_i, P_j) - b_k$
10     $B(P_i) \leftarrow B(P_i) \cup b_k$
11     if $P_j$ is a BitTorrent client then
12       $P_i$ stores $b_k$
13     else if $P_j$ is a PACOM client then
14       if $b_k$ is not a dummy block then
15         Decode $d_k$ from $b_k$
16         if $d_k$ is required to be forwarded then
17           $P_x \leftarrow$ The next hop of $d_k$
18           Put $d_k$ into $Q(P_i, P_x)$
19   //Receiving request and responding
20   if $P_i$ receives a request from $P_j$ for a block $b_k$ then
21     if $P_j$ is a BitTorrent client then
22       $P_i$ sends $b_k$ to $P_j$
23     else if $P_j$ is a PACOM client then
24       if $Q(P_i, P_j)$ is not empty then
25         $d \leftarrow$ Dequeue $Q(P_i, P_j)$
26       else
27         $d \leftarrow$ dummy data
28       $P_i$ embeds $d$ into $b_k$ and sends to $P_j$
```

Theorem 1 confirms the essential power of PACOM against the traffic analysis attacks. It points out that no statistical correlation can be built between the input link and output link of any PACOM client in a communication path. Corollary 1 further shows that the traffic patterns of the links in a communication path are all independent. Thus, the traffic analysis attacks are unable to build any correlation among the links on the path. More discussions about the anonymity analysis of PACOM will be presented in Section 4.

**Theorem 1.** *The traffic patterns of the input link and output link of any PACOM client in a concealed communication path are independent.*

**Proof.** Suppose that $P_i, P_j, P_k$ are three adjacent PACOM clients and $P_i \rightarrow P_j \rightarrow P_k$ is a path to transfer the embedded communication data. We now prove that the traffic in the input link $P_i \rightarrow P_j$ is independent of the output link $P_j \rightarrow P_k$. According to Algorithm 1, the traffic of any link contains the request packet flows and response packet flows. The traffic pattern of the link $P_i \rightarrow P_j$ is denoted as a tuple $F_{ij} = (Q_{ij}, R_{ij})$, where $Q_{ij}$ and $R_{ij}$ are the traffic patterns of the request and the response packet flows in the link, respectively. Similarly, the traffic pattern of $P_j \rightarrow P_k$ is defined as $F_{jk} = (Q_{jk}, R_{jk})$. Taking the traffic pattern as a random variable, we have the join probability deduction:

$$Pr(F_{ij}, F_{jk}) = Pr(Q_{ij}, R_{ij}, Q_{jk}, R_{jk}) \tag{1}$$

As described in Algorithm 1, the response packet flow is only driven by the arrival of requests, but has no relationship with the embedded data. Thus, $R_{ij}$ can be presented as a function of the request flow $Q_{ji}$: $R_{ij} = f(Q_{ji})$. Similarly, we have $R_{jk} = f(Q_{kj})$. From Eq. (1), we have:

$$Pr(F_{ij}, F_{jk}) = Pr(Q_{ij}, f(Q_{ji}), Q_{jk}, f(Q_{kj})) \tag{2}$$

Because each client sends the block requests independently of each other, different request packet flows are statistically independent. Thus, from Eq. (2) we can infer that:

$$\begin{aligned} Pr(F_{ij}, F_{jk}) &= Pr(Q_{ij}, f(Q_{ji})) \cdot Pr(Q_{jk}, f(Q_{kj})) \\ &= Pr(Q_{ij}, R_{ij}) \cdot Pr(Q_{jk}, R_{jk}) \\ &= Pr(F_{ij}) \cdot Pr(F_{jk}) \qquad \square \end{aligned} \tag{3}$$

**Corollary 1.** *The traffic patterns of any pair of different links belonging to the same communication path are independent of each other.*

The proof of Corollary 1 is similar with Theorem 1.

### *3.3. Client's joining the PACOM network*

A portion of PACOM clients in the network act as the boostrapping nodes, which share files, create PACOM swarms, and guide new arrived PACOM clients to join the swarms. For clarity, the file shared by a boostrapping node is named as *PACOM file* in this paper. For any boostrapping node $P_b$, the *info_hash* of each PACOM file on $P_b$ is assigned as:

$$I_x = H_F(x, IP(P_b), d) \quad (1 \leqslant x \leqslant \gamma) \tag{4}$$

Here $x$ means the local ID of the No. $x$ PACOM file shared by $P_b$ at the date $d$. $\gamma$ is the number of PACOM files shared by $P_b$, which can vary in different boostrapping nodes. According to the observation [44] about the distribution of the number of transferring files on each BitTorrent client, $\gamma$ is set to follow the geometric distribution $Pr(\gamma) = p^{\gamma-1}(1-p),\ (1 \leqslant \gamma \leqslant 100)$, where $p$ is set to 0.8551 as suggested by [44]. $H_F$ is a uniform hashing function to map the conjunction of the parameters into a 20-byte number, $IP(P_b)$ means the IP address of $P_b$, and $d$ is the current date. By invoking the BitTorrent API *announce_peer*$(I_x)$, $P_b$ announces to share the No. $x$ PACOM file and create the corresponding No. $x$ PACOM swarm on $P_b$.

For the sake of guiding other PACOM clients to download the PACOM file, $P_b$ hides the PACOM file ID $x$ into sending-out BitTorrent messages while downloading another popular file $F_i$ shared in the BitTorrent network. The detailed information hiding procedure is presented in Appendix A. For any newly joining client $P_i$, it also joins the BitTorrent network and downloads $F_i$. While $P_i$ exchanges BitTorrent messages with the other clients in the BitTorrent swarm transferring $F_i$, it can receive messages from $P_b$ and decode the ID $x$ as described in Appendix A. $P_i$ then calculates $I_x$ according to Eq. (4), invokes the BitTorrent API *get_peers*$(I_x)$ to get the information of the clients in the No.$x$ PACOM swarm created by $P_b$, and joins the swarm.

In order to make the distribution of the size of PACOM swarms similar with other normal BitTorrent swarms, we set a variable threshold $\beta$ of the maximum size of a PACOM swarm. When the number of the clients in a PACOM swarm is larger than $\beta$, we announce that the swarm is full. Based on the observation [48] about the distribution of the swarm size in the BitTorrent network, $\beta$ is set to follow a Pareto distribution with the mean value as 11.12 and the variance as 13.42. While considering this threshold, after $P_i$ joins the PACOM swarm created by $P_b$, there are two cases considered by $P_b$ as follows:

*Case 1:* If all PACOM swarms created by $P_b$ are full, $P_b$ stops serving as a boostrapping node by stopping the information hiding.

*Case 2:* Otherwise, $P_b$ randomly selects a PACOM swarm $S_k$ from the ones created by $P_b$ and not full. $P_b$ then hides the ID of $S_k$ into the BitTorrent messages. The following new arrived clients contacting with $P_b$ will be guided to join $S_k$.

Furthermore, if a newly joining client detects more than one boostrapping nodes, it can join multiple PACOM swarms for better communication performance. In the PACOM network, the number of bootstrapping nodes is designed to be within a predetermined threshold. If a newly joining client finds that the number of bootstrapping nodes is smaller than the threshold, it makes itself a bootstrapping node. On the other hand, when a bootstrapping node is in the case 1 above, it stops serving as a bootstrapping node.

### *3.4. Neighbor-to-neighbor communication channel*

After a client joins a PACOM swarm, it starts to download the PACOM file shared in the swarm and transfer the file blocks embedding communication data with its neighbors as described in Algorithm 1. All messages involved conform to the BitTorrent protocol. In this way, the neighbor-to-neighbor communication channels can be built between adjacent PACOM clients in the network.

The simplest way to embed the data in a PACOM file block is just to construct the block as a encrypted communication data chunk with the same size like a common BitTorrent file block. This method is easy to deploy but adversaries can distinguish the PACOM file blocks from other normal BitTorrent blocks through statistical analysis of the block content. A more reliable solution is to adopt the steganography method to hide communication data into file blocks. A broad spectrum of steganography methods [46] can be used here to hide data in different types of files, such as video, audio and text documents.

In order to avoid content based correlation analysis from adversaries, the communication data should be encrypted before being embedded into the file blocks. For this purpose, we design an initialization procedure for the communication channel to negotiate a symmetric encryption key before transferring data. The procedure is illustrated in Fig. 4. $P_i$ and $P_k$ are two adjacent clients download a PACOM file in a same PACOM swarm. Following the BitTorrent protocol, they send requests to each other and exchanges the file blocks. When $P_i$ receives a request for a block, it finds the corresponding block in its local buffer, embeds the information $\{g^x\}$ within, and responds with the block. Here $g^x$ is the first half of the Diffe–Hellman handshake [47]. After a similar procedure, $P_k$ responds to $P_i$ with another file block embedding $\{g^y\}$, where $g^y$ is the other half of the Diff–Hellman handshake. Till now, both clients have reached an agreement on using $g^{xy}$ as the symmetric encryption key.

After the initialization of the channel, $P_i$ and $P_k$ embed the communication data $E_{ik}\{len, data\}$ into each transferred file block. Here $E_{ik}$ means the data in the following braces is encrypted by using the AES algorithm with the symmetric key $g^{xy}$ between $P_i$ and $P_k$, *data* is the communication data, and *len* is the length of the data. If a client has no communication data to send when receiving a file block request, it responds with a file block embedding dummy data as $E_{ik}\{0, pad\}$, where *pad* is a randomly constructed padding.

The file blocks containing dummy data transferred in the channel composes the cover traffic, which are used to shape the request driven traffic pattern like BitTorrent protocol. The volume of the cover traffic is adaptive to the variation of the communication traffic in the network. The more communication data transferred in the PACOM network, the less cover traffic is produced. The flexibility of the cover traffic reduces its side effect on the throughput of the PACOM network. Meanwhile, the cover traffic is confined in the communication between neighbors in the same PACOM swarm and does not flood out to affect other clients in different swarms.

Moreover, according to Algorithm 1, the maximum volume of communication data that can be received by a client from its neighbors in a PACOM swarm is determined by the size of the file shared in the swarm. When the client finishes downloading a file from a PACOM swarm, it cannot receive communication data from the swarm any more. At that time, the client rejoins the PACOM network to take part in another PACOM swarm to keep its communication channels open.

### 3.5. End-to-end anonymous communication

Based on the neighbor-to-neighbor communication channels, the end-to-end circuits between any pair of PACOM clients can be built to support anonymous communication. As illustrated in Fig. 5, the circuits span different swarms and link multiple neighbor-to-neighbor communication channels. The communication in a circuit may pass several hops of clients, and the data transferred between any pair of neighboring clients are based on the neighbor-to-neighbor channel, which provides data encryption and traffic transformation services as mentioned in the last section. In this way, all circuit messages are encrypted and embedded in the transferred file blocks, whose format conforms to the BitTorrent protocol.

The setup procedure of a circuit is illustrated in Fig. 6. $P_s$ and $P_t$ are two clients in the PACOM network. When $P_s$ plans to build a circuit to $P_t$, $P_s$ broadcasts the handshake

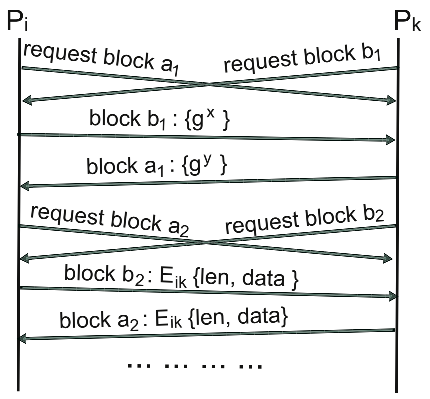

**Fig. 4.** Neighbor-to-neighbor communication procedure.

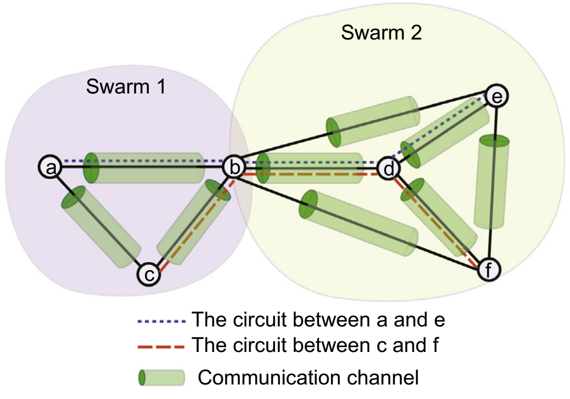

**Fig. 5.** The circuits are built to support end-to-end anonymous communication. The clients a, b, c are in the PACOM swarm 1, and b, d, e, f are in the swarm 2. There are two circuits constructed in the network.

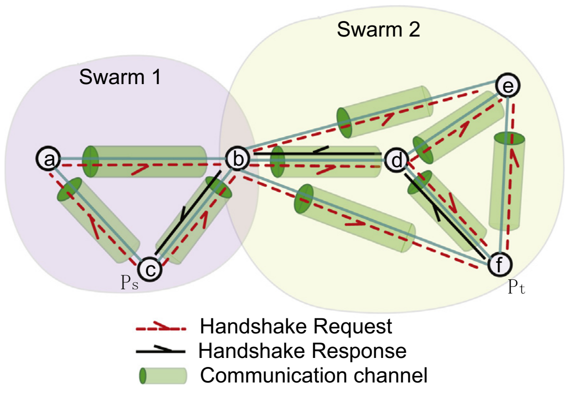

**Fig. 6.** The setup procedure of a circuit between $P_s$ and $P_t$.

request message to all PACOM neighbors through its communication channels:

$$\{type = 0, h\_id, TTL, E_t\{h\_id, time, sec\_str, K\}\} \quad (5)$$

Here *type* is the type of the message and is set to zero to indicate a handshake request. *h_id* is the unique identity of the message generated by $P_s$. $E_t\{\ldots\}$ means using the RSA algorithm to encrypt the data in the braces with the public key of $P_t$, which is obtained by $P_s$ from CA services or in some other private way. *time* is the sending time of the message, which is designed to prevent the relay attacks. *sec_str* is a secret string generated by $P_s$. $K$ is a random key generated by $P_s$ to encrypt and decrypt the flowing end-to-end communication between $P_s$ and $P_t$. *TTL* is an integer indicating how long the message can survive in the network.

Each PACOM client receiving the handshake request checks the value of the *TTL* in the request. If *TTL* is equal to 0, the client discards the request. Otherwise, it decreases *TTL* by one, and forwards the modified request to all its PACOM neighbors. It also tries to decrypt the encrypted portion of the request by using its private key. It can judge whether the decryption is successful by checking whether the two *h_id* field in the decrypted message is equal. No one except $P_t$ in the network can decrypt the message successfully. Moreover, while forwarding a handshake request, the client also puts down a circuit record $(last\_hop, h\_id, next\_hop, r\_time, suc\_flag)$. Here *last_hop* is the IP address of the host, which the handshake is received from, and *next_hop* is corresponding to the host, which the handshake is forwarded to. *r_time* is the time of receiving the handshake. *suc_flag* indicates whether the circuit is built successfully and it is initialized to be 'false'. All circuit records created by a client constitute its *circuits table* used for routing messages in circuits. Each client checks its circuits table periodically and purges those old records, in which *suc_flag* is false and the time since *r_time* is longer than a threshold *t*. *t* can be set to the longest possible waiting time for a circuit to be built.

If $P_t$ receives the handshake request from another peer $P_i$, it constructs the handshake response as:

$$\{type = 1, h\_id, E_K\{sec\_str\}\} \quad (6)$$

Here $E_K\{\ldots\}$ means using the AES algorithm to encrypt the data in the braces with the key $K$ received from the handshake request. *sec_str* and *h_id* are both the same as those in the request. $P_t$ sends the response backwards to $P_i$ through the communication channel. Then $P_i$ searches its circuits table for the record which has the identity as *h_id* and the *next_hop* as $P_t$. $P_i$ changes the *suc_flag* of the record to 'true' to indicate successful construction of the circuit, and sends the response to the host *last_hop*. In a similar manner, the intermediary clients receiving the response send it hop by hop along the reversed path of where the handshake request was forwarded before, until it reaches $P_s$.

After $P_s$ receives the response, the circuit is ready for $P_s$ and $P_t$ to start end-to-end communication. When $P_s$ plans to send data to $P_t$, it constructs the message as:

$$\{type = 2, h\_id, E_K\{d\_id, MD5, len, data\}\} \quad (7)$$

Here *type* is set to 2 to indicate this is a data packet. *h_id* is the identity of the used circuit. *d_id* is the identity of the data packet. *data* is the data payload. *len* is the length of the data. *MD5* is the MD5 checksum of *data*, which is designed to preserve the integrity of the data. $E_K\{\ldots\}$ means using the AES algorithm to encrypt the data in the braces with the key $K$ generated in the handshake phase. $K$ is only known by $P_s$ and $P_t$, so no other ones can achieve the encrypted checksum and data. This makes it impossible for the ones except for $P_s$ and $P_t$ to crack the checksum by manipulating the data to get the same checksum. Thus the MD5 checksum is strong enough to be used here to protect the integrity of data.

While transferring the message, $P_s$ and the intermediary clients forward it to the *next_hop* of the circuit by checking their circuit tables. Through the circuit, the message may reach $P_t$ in the end. In a similar way, $P_t$ can send data to $P_s$ through the circuit in the reversed direction. $P_t$ constructs the message with the same format as what $P_s$ sends.

Similarly with most mix systems [1–3], PACOM preserves the anonymity of $P_s$, the source of the circuit. The target $P_t$ and the intermediary clients in the circuit only know the last hop in the circuit while having no idea about the source.

It is also easy to extend the communication protocol to further optimize the performance of PACOM in the following aspects:

(1) $P_s$ can setup multiple circuits to $P_t$ to improve the reliability and efficiency of the communication. The data packets sent from $P_s$ to $P_t$ can be transferred in parallel in multiple circuits.
(2) The length of the circuit can be extended to promote the anonymity in the following way. In the handshaking phase, $P_s$ can only forward the request to one randomly selected neighbor. Each client receiving the request forwards it to one randomly selected neighbor with a probability $p_0$ while broadcasting it to all neighbors with the probability $1 - p_0$. A larger $p_0$ will lead to a longer circuit constructed between $P_s$ and $P_t$.

(3) Some acknowledgment mechanism can be added to guarantee the reliability of the communication. When the number of the received data packets from a circuit reaches a threshold, the receiver sends back the acknowledgement information (ACK) about what data packets it has received. If the sender has not received the ACK for a certain period of time, it discards the circuit and rebuilds a new one.

(4) The flooding of the handshake request can be utilized by the malicious clients to launch overloaded attacks in the network. Without any restriction, the traffic of malicious requests will consume most of the capacity of the communication channels. This unexpected situation can be avoided by adding a threshold to limit the bandwidth used for transferring handshake requests in the channels. The request traffic exceeding the threshold may be simply discarded to avoid further spreading.

## 4. Anonymity analysis

In this section, we discuss the anonymity model of PACOM, and analyse the effectiveness of PACOM against various attacks under the threat models presented in Section 3.1.

### 4.1. Anonymity model of PACOM

We model the anonymity of PACOM in terms of the *effective anonymity set size*, which is an information theoretic metric defined by Serjantov et al. [26]. The effective anonymity set size $S$ of an anonymous communication system is calculated as the entropy of the anonymity probability distribution:

$$S = -\sum_{P_u} Pr(P_u)\log(Pr(P_u)) \tag{8}$$

Here $Pr(P_u)$ denotes the probability of the client $P_u$ to be the information sender from the view of a adversary. A larger $S$ means higher anonymity.

### 4.2. Protocol testing attacks

The degree of similarity between a PACOM client and a normal BitTorrent client decides whether they can be mixed together to confuse adversaries. Recently, Houmansadr et al. propose the attacks [54] to identify mimic network flows by testing each detailed implementation issue of the network protocol such as reaction to errors and network conditions. The research [54] also points out that one promising way to resist such attacks is not to mimic, but to run the actual protocol and hide data in the genuine flows. The design principle of PACOM is consistent with this opinion. As illustrated in Fig. 2, each PACOM client adopts the BitTorrent module [50] to run the BitTorrent protocol and transfer file blocks containing communication data. The traffic of each link is triggered by the BitTorrent protocol.

Table 2 summarizes the behaviors of a PACOM client. The table shows that all traffic of a PACOM client is produced by the BitTorrent module and exactly consistent with the BitTorrent protocol. The only chance for adversaries to distinguish PACOM flows from BitTorrent flows is to crack the steganography mechanism.

As defined in Threat Model I of Section 3.1, adversaries cannot compromise PACOM clients in the Private Use Case and crack the steganography techniques. Thus, in this case, PACOM clients are indistinguishable from other BitTorrent clients. For a PACOM network parasitic in a BitTorrent network having $N$ online clients, adversaries determine the probability of each BitTorrent client being a communicating PACOM client as $1/N$. Thus the effective anonymity set size of the system in the Private Use Case is $\log N$ according to Eq. (8). While there are millions of BitTorrent users online in the Internet, the anonymity can be extended enormously.

On the other hand, while considering the Public Use Case of PACOM, adversaries are able to compromise a fraction of PACOM clients as defined in Threat Model II. After running for a long time, malicious clients can monitor the boostrapping nodes and join as many PACOM swarms as possible to locate most of the PACOM clients in the network. In this extreme scenario, the effective anonymity set size of the system changes to $\log n$, where $n$ is the total number of online PACOM clients. While PACOM is designed for public use in this use case, we can increase the number of PACOM users to increase the anonymity as what Tor [2] has done.

### 4.3. Traffic analysis attacks

The traffic analysis attacks are conducted by the adversaries to observe the network flow information such as the packet count, message volume, and time interval, and to build timing correlations between different links in the communication path. The adversaries of both Thread Model I and II can launch such attacks. The recently presented researches [21,22] point out that the existing flow transformation strategies fail to eliminate the timing correlation and prevent this kind of attacks.

The traffic analysis attacks can be classified into two categories roughly: active attacks and passive attacks. In the passive attack [21], the adversaries only observe the packet flow passively and analyse the timing pattern to determine the correlation among the links. The traffic pattern of a link is modeled as a pattern vector:

$$X_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,q}) \tag{9}$$

Here $x_{i,k}$ $(1 \leqslant k \leqslant q)$ is the number of packets in the $k$-th time interval. Then the correlation between two links is calculated as the distance between their pattern vectors in the frequency domain as follows:

$$d(X_i, X_j) = \frac{\sqrt{\langle X_j^F, X_j^F \rangle}}{\langle X_i^F, X_j^F \rangle} \tag{10}$$

Here $X_i^F$ is the frequency spectrum of $X_i$ calculated by using the FFT or Wavelet algorithm. The operator $\langle \cdot, \cdot \rangle$ denotes the inner product of two vectors. The links with a smaller distance between their pattern vectors should have stronger correlation. By calculating the distance between the observed links, the adversaries can correlate the links on a communication path.

**Table 2**
Behavior of a PACOM client.

| Behavior | Deployment |
|---|---|
| *Join the network* | |
| S1: Download some popular files to join the BitTorrent network (Section 3.3) | Use the BitTorrent module to download files |
| S2: If a PACOM client performs as a boostrapping node, it shares files (Section 3.3) | Use the BitTorrent module to share files |
| S3: Download files shared by some PACOM clients (Section 3.3) | Use the BitTorrent module to transfer file blocks embedding communication data |
| *Communication* | |
| S4: Build neighbor-to-neighbor channels (Section 3.4) S5: Construct circuits (Section 3.5) S6: Communication in circuits (Section 3.5) | All links of a PACOM client are maintained by the BitTorrent module and used to transfer file blocks embedding communication data |
| S7: Reaction to network churn S8: Reaction to error | It is handled by the BitTorrent module which follows the BitTorrent Protocol |

On the other hand, in the active attacks [22–24], the adversaries try to transform the attacked links by tuning the time interval or the count of packets, and then observe how the traffic of other links is affected. The most correlated links observed are very likely to be on the same communication path with the attacked one.

PACOM is able to resist both passive and active traffic analysis attacks. Specifically, according to Theorem 1 and Corollary 1, the traffic patterns of all PACOM links are independent with each other. The distance (measured in Eq. (10)) between two links on a communication path is not necessarily smaller than the distance of other ones, so the passive traffic analysis fail to correlate the links. Meanwhile, because the independent traffic pattern of PACOM, manipulating the traffic on some PACOM links have no influence on other ones. Thus PACOM is also immune to the active attacks.

### 4.4. Other attacks

Besides the traffic analysis attacks, below we analyse how PACOM resists some other common attacks against anonymous communication systems in this section.

#### 4.4.1. Disclosure attacks

Recently, some statistical disclosure attacks [36,55,56] are proposed to de-anonymize mix networks. These attacks consider the whole communication system as a big black box and observe the input messages sent from each user and the output messages received by each one. Aided by some statistical calculation, the friendship among users can be discovered with a high probability after observing the traffic for a long time.

A basic assumption of the statistical disclosure attacks [36,55,56] is that adversaries can observe the exact numbers of messages sent and received by each client at each round, which form the basic inputs of the attacks. In PACOM, the messages mixed with dummy data are embedded into file blocks as mentioned in Section 3.2. Because all embedded data is encrypted, adversaries cannot determine which file block contains PACOM data or just dummy data. Thus they cannot determine the precise number of data packets transferred by a client at any timing interval.

Another basic assumption of the attacks is that the observed traffic of each client should be related to its transferring data. Different from traditional mix networks, the traffic on each PACOM link is not related to the arrival time and volume of the PACOM data handled by each client, but triggered by the BitTorrent file sharing protocol.

Based on above analysis, statistical disclosure attacks [36,55,56] are not effective to calculate user profiles in PACOM.

Another variant of discourse attacks is the intersection attack [57] proposed earlier, which determines the communication partner of a user by intersecting the anonymity sets of his sent messages. Furthermore, Wright et al. [58] point out that the P2P anonymous communication systems can be particularly vulnerable to this kind of attacks due to their dynamic network situation. The attacker can periodically put down the list of online clients when the information sender contacts the receiver, and calculate the intersection of the lists to obtain a much smaller set of clients containing the sender. Due to the dynamic membership of the P2P networks, it is very possible to narrow down the anonymity set efficiently in this way. The experiments [58] based on simulated P2P traffic and real Tor logs all show that the success rate of such attacks can approach 1 if the attacks last for a enough long time. While all PACOM clients are running in the BitTorrent network, which is a typical P2P file sharing network, the attacker can launch such attacks to identity the information sender by analyzing the uptime statistic of all BitTorrent clients. Attackers of both Threat Model I and II are able to perform this kind of attacks as illustrated in Fig. 7. In this example, two PACOM clients $P_A$ and $P_B$ join the BitTorrent network together with other three BitTorrent clients $P_i$ $(1 \leqslant i \leqslant 3)$. The attacker tries to determine which client is communicating with $P_B$ by calculating the intersection of the lists of clients online with $P_B$. Fig. 7 shows a successful attack to identify $P_A$.

We consider the protection strategies against the intersection attacks in the following two cases:

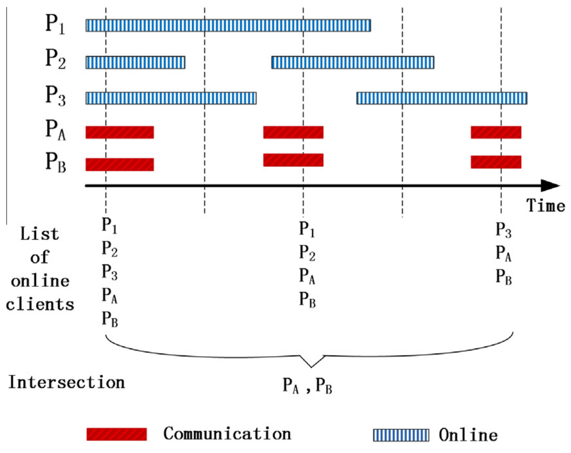

**Fig. 7.** Intersection attack against PACOM. $P_A$ and $P_B$ are both PACOM clients. $P_A$ is the information sender and $P_B$ is the receiver. $P_i$ $(1 \leqslant i \leqslant 3)$ are BitTorrent clients. The attacker tries to identify who is communicating with $P_B$.

- *Innocent Receiver*. In this case, the information receiver is not malicious and does not cooperate with the attacker to identify the information sender.
- *Malicious Receiver*. In this case, the information receiver is malicious and tries to identify the information sender.

For the *Innocent Receiver* case, the attacker can observe the traffic of the receiver, but cannot determine which packet transferred by the receiver contains communication data or just dummy data. Thus, the attacker cannot judge whether the receiver is communicating or not. To resist the intersection attacks, PACOM clients can extend their uptime to confuse the attacker even they are not communicating. Specifically, each PACOM client can join the network at a randomly selected time before it gets ready to communicate and leave the network at another randomly selected time after the end of the communication. Each client can also join and leave the network randomly when

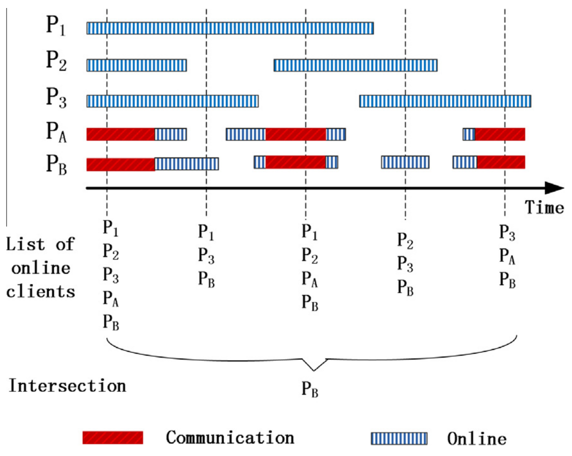

**Fig. 8.** Extend the uptime of PACOM clients to resist the intersection attacks in the *Innocent Receiver* case. $P_A$ and $P_B$ are both PACOM clients. $P_A$ is information sender and $P_B$ is receiver. $P_i$ $(1 \leqslant i \leqslant 3)$ are BitTorrent clients. The attacker tries to identify who is communicating with $P_B$.

no communication happens. Fig. 8 shows this strategy is effective to resist the attack. The extension of uptime makes $P_A$ not always online with $P_B$. Thus, $P_A$ may probably not exist in the intersection of the online clients and the attacker has no idea to identify who is communicating with $P_B$ in this case.

We also conduct some experiments to simulate the attack based on the real BitTorrent logs [59], which are collected by University of Massachusetts and published online [60]. The logs contain the snapshots of 845,014 file downloading sessions from 10/27/2003 to 1/16/2004. The snapshots are collected every 30 min, each of which records the IDs of online BitTorrent clients. We use the logs to simulate an intersection attack. Two PACOM client $P_A$ and $P_B$ are parasitic in the network. $P_A$ initiates a communication session to $P_B$ every day since 10/28/2003. Each session starts at a randomly selected time and lasts for $h$ $(1 \leqslant h \leqslant 10)$ hours. We simulate the intersection attack to identify which client is communicating with $P_B$. We record the list of online clients every one hour during the uptime of $P_B$. Then we calculate the intersection of the lists to achieve a smaller suspicious set. If the set contains $P_A$ and its size is within 10, we announce a successful attack as [58]. We repeat the experiment for 100 times to calculate the probability of attacker success. Fig. 9(a) shows the relation between the running time and the probability of attacker success. We can see that the random extension of uptime can effectively resist the attack as expected.

For the *Malicious Receiver* case, the receiver knows the exact time when the sender is communicating, so the strategy to extend uptime may be ineffective against the attack. On the other hand, since the attack [58] is based on the intersection of client identities such as IP addresses or user IDs, we can change the sender's identity periodically to resist such attacks. That can make the identity of the sender out of the intersection set, which is achieved by the attacker. In real-world deployment, the user ID in the BitTorrent network can be easily changed, since it is randomly assigned by each client. Meanwhile, the changing of IP address is also feasible. For the users of broadband cable and ADSL which are popularly deployed, the IP addresses are dynamic by default. We can reconnect to the network to achieve a different IP address. Moreover, we can make use of some proxy servers to change IP address dynamically. In this protection strategy, how frequent a PACOM client should change its identity to effectively resist the intersection attacks is a critical problem to be considered. We conduct some experiments based on the same BitTorrent logs as above to verify the choice of the parameter. In this experiment, two PACOM client $P_A$ and $P_B$ keep online and communicate since 10/28/2003. The attacker tries to identify which one is communicating with $P_B$. To resist the attack, $P_A$ changes its identity every $\xi$ hours since it joins the network. Fig. 9(b) illustrates the relation between $\xi$ and the probability of attacker success. The graph shows that $P_A$ can escape from the attack if it can change its identity within 70 h. That may be applicable in real-world deployment of the communication system, since most of the communication sessions can be over within several hours. The changing of the identity may affect the communication efficiency slightly.

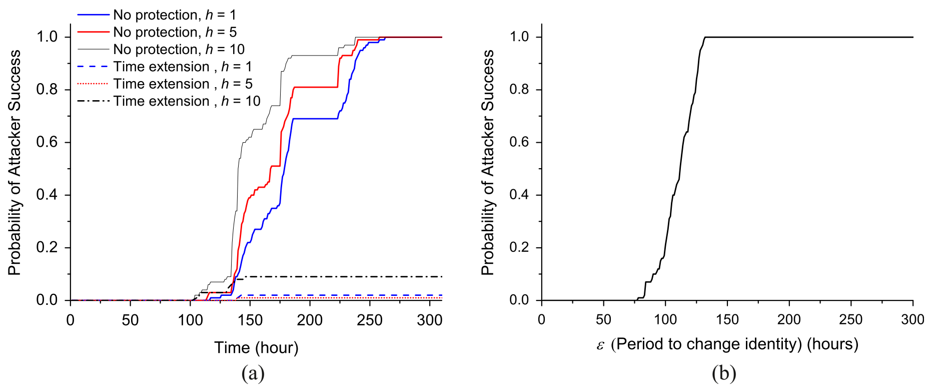

**Fig. 9.** The probability of attacker success in different cases. (a) The intersection attack in the *Innocent Receiver* case. Here 'No protection' means the basic PACOM model, and 'Time extension' means the PACOM equipped with the strategy to randomly extend the uptime of PACOM clients. (b) The intersection attack in the *Malicious Receiver* case. $\xi$ means the period to change the identity of a PACOM client.

### *4.4.2. Collaborating attacks*

As defined in Section 3.1, the adversaries of Threat Model II can compromise some fraction of PACOM clients in the network. The malicious clients can disguise themselves as boostrapping nodes and create PACOM swarms containing only malicious clients. While a normal PACOM client $P_i$ joins the malicious swarm, its communication with the malicious neighbors in the swarm can be tracked. This kind of attacks are named as *collaborating attacks*.

Specifically, a client is called to be in the *trap-mode* when the PACOM swarms containing the client are all malicious swarms. When observing that a client in the trap-mode sends out a message but has not received a message before, adversaries can judge it as an information sender. However, for a client not in the trap-mode, adversaries cannot make such a decision and the probability for the client to be a communication participant is equal to that of any PACOM client else.

Based on the above analysis, we have the following theorem about the effectiveness of PACOM against the collaborating attacks:

**Theorem 2.** *The effective anonymous set size of PACOM against the collaborating attacks is equal to* $log(n(1-\rho^r))$. *Here r is the average number of PACOM swarms joined by each client, ρ is the ratio of malicious boostrapping nodes, and n is the total number of PACOM clients.*

The proof of Theorem 2 is presented in Appendix B. From Theorem 2, we can see that the higher anonymity can be achieved by increasing $r$, the average number of the PACOM swarms joined by a client. On the other hand, higher $r$ will cause higher bandwidth consumed for the client. It is a trade-off between security and performance to tune the value of $r$.

## 5. Discussion on the deployment

As mentioned in Section 3.4, there are two basic strategies to embed communication data into transferred file blocks. The first simple method is to construct a block as one segment of encrypted communication data with the same size as a BitTorrent file block. The other advanced method is to adopt steganography mechanisms to hide data into file blocks. The strength of steganography is that the PACOM traffic is more indistinguishable from normal BitTorrent traffic, while the weakness is that the bandwidth of communication can be relatively low. Thus whether to adopt steganography is related to the trade-off between security and performance.

For the users in the Private Use Case, it is hard for adversaries to compromise a PACOM client, as described in Threat Model I. PACOM is designed to provide strong anonymity protection by mixing PACOM clients and millions of BitTorrent clients. Steganography is necessary in this case to confuse adversaries. However, the situation in the Public Use Case is different. The PACOM client is designed for public use and the number of online PACOM users can be much larger than that in the Private Use Case. In this case, adversaries have power as defined in Threat Model II to compromise some faction of PACOM clients to monitor the network. After observing for a long time, adversaries can even detect most of the PACOM clients in the network as analyzed in Section 4.2. In this case, steganography is not effective to confuse adversaries any longer, and the anonymity of PACOM is only related to $n$, which is the number of online PACOM clients. Thus, in the Public Use Case, we can just use the simple method to construct file blocks instead of steganography. According to the analysis in Section 4.3, PACOM in this configuration is still effective to resist traffic analysis, because the independent traffic pattern of each link is not affected and still triggered by the BitTorrent protocol.

## 6. Performance evaluation

In this section, by conducting simulation experiments, we observe how the topology of the PACOM network affects the performance of the anonymous communication. Meanwhile, through emulation experiments, we validate the effectiveness and efficiency of PACOM against the traffic analysis attack in real network environments.

### *6.1. Simulation setup*

To observe the topology of the PACOM network, we simulate multiple PACOM networks with the number of clients scaling from 32 to 8192. Each client joins multiple PACOM swarms concurrently. The configuration of the PACOM swarms is consistent with the features observed from the swarms in the BitTorrent network. The total number $\gamma$ of the PACOM swarms joined by a client is set to follow the geometric distribution $Pr(\gamma) = p^{\gamma-1}(1-p),\ (\gamma \geqslant 1)$ according to the observation [44] on the BitTorrent network. $p$ is set to 0.8551 following the observation in [44]. Meanwhile, based on the observation [48] over the BitTorrent dataset containing 1,682,355 swarms, the size of the PACOM swarms $\beta$ is set to follow a Pareto distribution with the mean value as 11.12 and the variance as 13.42. Considering the limitation of the available bandwidth on each client, the maximum number of connections each client can build is set to less than a threshold as 100.

## 6.2. Metric of simulation

We mainly use *average shortest path length* to measure the topology of the PACOM network. According to the end-to-end anonymous communication mechanism of PACOM introduced in Section 3, the length of the circuit between two clients is larger than or equal to the length of the shortest path between them. Thus, the average shortest path length is a very important factor related to the efficiency of the anonymous communication.

## 6.3. Simulation results

In the experiment, we start from an empty network, and construct the network by arranging the clients to join the network one by one. Fig. 10 shows how the average shortest path length rises as the network scale increases, and how the dynamic behavior of clients affects the shortest path length. '0% rejoin' means all clients join the network one by one, without any one leaving the network. In this scenario, the shortest path length grows linearly with the number of clients. This undesired situation can be changed by the dynamic leaving and joining behaviors of clients. As illustrated in Fig. 10, while more than 5% of the clients rejoin the network, the average shortest path length grows logarithmically in terms of the number of the clients, and reaches less than 4 in the PACOM network with 8192 clients. Meanwhile, the average shortest path length decreases when more clients rejoin the network.

From the observation above, we can see that the network dynamics can promote the connectivity and reduce the average shortest path length. The dynamics can happen in the PACOM network naturally or artificially in the following cases: (1) the PACOM clients are shut down or restarted by the users; (2) the PACOM clients rejoin when finishing downloading a file in a PACOM swarm; and (3) we can force the clients to rejoin network periodically.

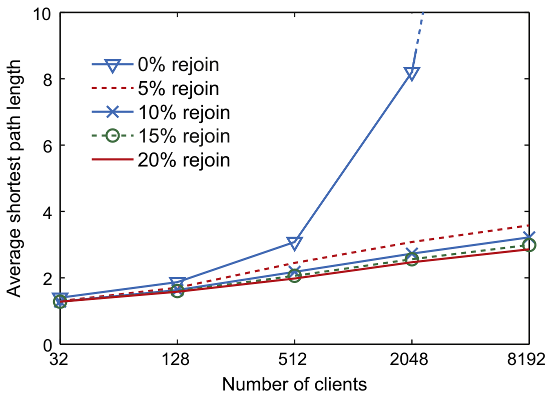

**Fig. 10.** Average shortest path length of the PACOM network. The number of boostrapping nodes is set to 10 in this experiment.

## 6.4. Emulation setup

In the following experiments, we conduct emulations to validate the effectiveness of PACOM against the traffic analysis attacks in real network environments. We focus on the recently presented passive traffic analysis attack [21] in the experiments, while theoretically analyzing the other analysis attacks [22–24] in Section 4.

We implement a PACOM network with 8 clients, the topology of which is illustrated in Fig. 11. To emulate the real network situation, the route between any pair of clients is configured to pass the WANem [49] router, which is designed to emulate the WAN characteristics such as network delay, packet loss and jitter in a LAN environment. The network delay between any pair of clients is configured to be randomly distributed in [1 ms,100 ms]. $P_1 \sim P_5$ belongs to a PACOM swarm and $P_4 \sim P_8$ belongs to another one. Each client builds connections to others in the same swarm. We emulate the end-to-end anonymous communication between the clients and conduct the traffic analysis attack [21] in the network.

To validate the correctness of our implementation of the analysis attack and compare PACOM with traditional systems, we also implement the mix network similar to [21], which is illustrated in Fig. 12. The network delay between any pair of clients is also configured to be randomly distributed in [1 ms,100 ms].

## 6.5. Metrics of emulation

First, we use *detection rate* [21] as the major metric to measure the effectiveness of an anonymous communication system against the traffic analysis attacks. Detection

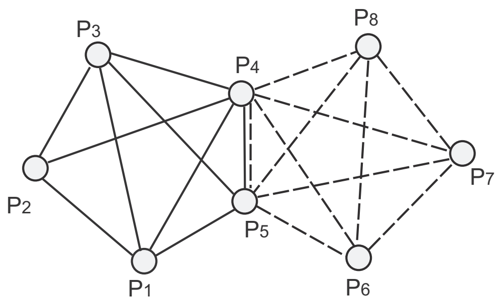

**Fig. 11.** The PACOM network contains 8 clients. $P_1, P_2, P_3, P_4$, and $P_5$ belong to a PACOM swarm. $P_4, P_5, P_6, P_7$ and $P_8$ belong to another PACOM swarm.

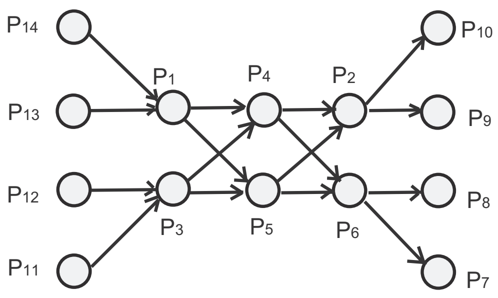

**Fig. 12.** The mix network with 14 clients.

rate here is defined as the ratio of the number of correct correlation detections to the number of attempts during the attacks. Moreover, we measure the efficiency of the cover traffic in terms of *cover traffic rate*, which is defined as the average ratio of bandwidth consumed by cover traffic on each PACOM client.

### 6.6. Effectiveness against traffic analysis attacks

Firstly, we perform the traffic analysis attack [21] on the mix network illustrated in Fig. 12. Four circuits for anonymous communication are built in the network as listed in Table 3. Each circuit contains 3 hops and there are two input links and two output links in each hop. The traffic analysis attack [21] can be implemented to correlate the input link with the output link of each hop in any circuit by calculating the distance of the traffic pattern vectors defined in Eq. (10). The output link with smallest distance to an input link is guessed to be on the same circuit with the input link. If the guess is right, it is called a correct correlation detection. By implementing the correlation detection hop by hop, adversaries can successfully deduce the source and destination of the circuit.

Without any prior knowledge, the detection rate of random guess on each hop is 0.5 in the mix network. Fig. 13 shows the average detection rate of the traffic analysis attack on each hop, which reaches 1.0 as the detection time is increased. This means the attack is successful in detecting the correlated output link and input link. The result is consistent with the observation in [21], which shows that an adversary can accurately determine the output link through the traffic statistical analysis and the detection rate can be as high as 100% as long as enough data is collected. By using FFT with 50 ms as the sampling interval, we also observe the power spectrum of the traffic in the first hop of a circuit. The result is illustrated in Fig. 14. We can see that the correlation of the input link and output link is very strong.

**Table 3**
Circuits in mix network.

| ID | Circuits |
|---|---|
| 1 | $P_{14} \rightarrow P_1 \rightarrow P_4 \rightarrow P_2 \rightarrow P_{10}$ |
| 2 | $P_{13} \rightarrow P_1 \rightarrow P_5 \rightarrow P_2 \rightarrow P_9$ |
| 3 | $P_{12} \rightarrow P_3 \rightarrow P_5 \rightarrow P_6 \rightarrow P_8$ |
| 4 | $P_{11} \rightarrow P_3 \rightarrow P_4 \rightarrow P_6 \rightarrow P_7$ |

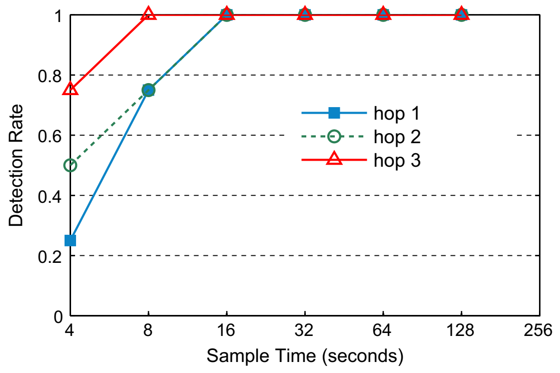

**Fig. 13.** Detection rate on each hop of a circuit in the mix network.

We conduct the same attack on the PACOM network illustrated in Fig. 11. In this network, we build 8 circuits as listed in Table 4. Each circuit also contains 3 hops, and each client appears in 5 circuits. The detection rate of the first hop is illustrated in Fig. 15. We can see that the detection rate is low and close to the detection rate in random guess, i.e., 1/3. This result confirms security analysis in Section 4.3, which announces the effectiveness of PACOM against traffic analysis attacks.

To observe the attack results more clearly, we conduct the experiment under a more rigorous condition. We keep only one of the 8 circuits in the PACOM network at a time and conduct the attack. In this case, there is only one circuit passing each client at most, so no mix of the circuits exists. This is the most favorable scenario for adversaries to identify the communication path. Fig. 16 shows the detection rate on each hop of the circuit in this case. As the detection time is increased, the detection rates drop to zero. This confirms that the attack fails in the PACOM network. We also observe the power spectrum of the traffic in the first hop of the circuit $P_3 \rightarrow P_1 \rightarrow P_5 \rightarrow P_8 \rightarrow P_7$ when the circuit is the only one in the network. The result is illustrated in Fig. 17. We can see that the correlation of

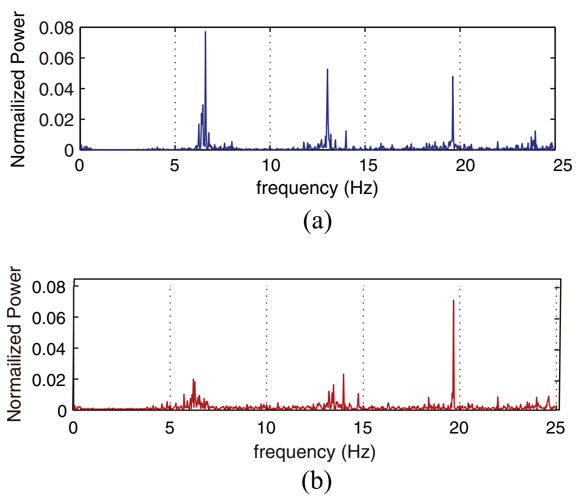

**Fig. 14.** Power spectrum of the traffic in the first hop of the circuit $P_{12} \rightarrow P_3 \rightarrow P_5 \rightarrow P_6 \rightarrow P_8$ in the mix network. (a) Power spectrum of input link $P_{12} \rightarrow P_3$. (b) Power spectrum of the output link $P_3 \rightarrow P_5$.

**Table 4**
Circuits in PACOM network.

| ID | Circuits |
|---|---|
| 1 | $P_2 \rightarrow P_1 \rightarrow P_5 \rightarrow P_6 \rightarrow P_7$ |
| 2 | $P_2 \rightarrow P_3 \rightarrow P_4 \rightarrow P_8 \rightarrow P_6$ |
| 3 | $P_3 \rightarrow P_2 \rightarrow P_4 \rightarrow P_7 \rightarrow P_8$ |
| 4 | $P_3 \rightarrow P_1 \rightarrow P_5 \rightarrow P_8 \rightarrow P_7$ |
| 5 | $P_1 \rightarrow P_3 \rightarrow P_5 \rightarrow P_6 \rightarrow P_8$ |
| 6 | $P_1 \rightarrow P_2 \rightarrow P_4 \rightarrow P_7 \rightarrow P_6$ |
| 7 | $P_4 \rightarrow P_7 \rightarrow P_6 \rightarrow P_8 \rightarrow P_5$ |
| 8 | $P_5 \rightarrow P_3 \rightarrow P_4 \rightarrow P_1 \rightarrow P_2$ |

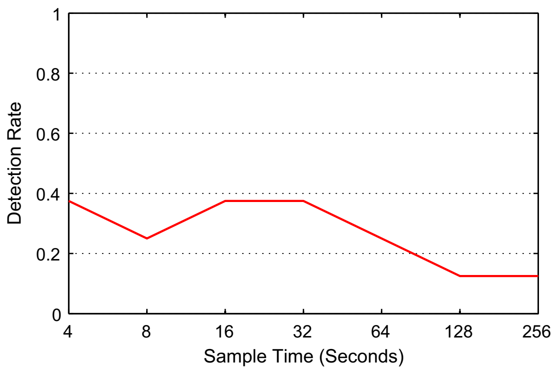

**Fig. 15.** Detection rate on the first hop of a circuit in the PACOM network with 8 circuits.

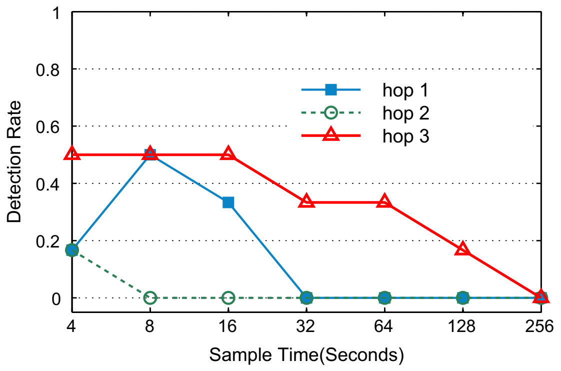

**Fig. 16.** Detection rate on each hop of a circuit in the PACOM network with only one circuit.

the input link and output link is not obvious compared with Fig. 14 in the mix network. This intuitively shows the independent traffic patterns between adjacent links in the PACOM network, which confirms the theoretical analysis in Theorem 1.

## 6.7. Efficiency of cover traffic

In this section, we measure the efficiency of the cover traffic in PACOM. We construct random circuits in the PACOM network, each of which is a randomly selected path with the same length of 4. Fig. 18 shows how much

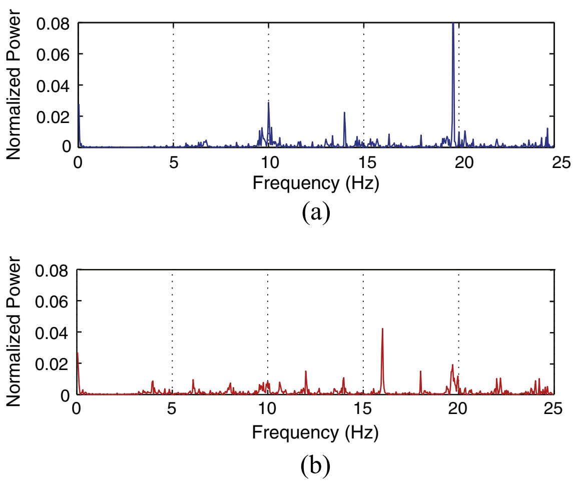

**Fig. 17.** Power spectrum of the traffic in the first hop of the circuit $P_3 \rightarrow P_1 \rightarrow P_5 \rightarrow P_8 \rightarrow P_7$ in the PACOM network. (a) Power spectrum of the input link $P_3 \rightarrow P_1$. (b) Power spectrum of the output link $P_1 \rightarrow P_5$.

bandwidth of each client is spent on the cover traffic as the number of circuits is increased on each client. We can infer that the cover traffic occupies about 50% of the bandwidth, when each client initiates one circuit to another one. As the number of circuits is increased, the cover traffic will drop correspondingly. The ratio drops to about 10% when each client initiates four circuits on average. This proves that the cover traffic is self-adaptive and not flooding in the PACOM network.

## 6.8. Prototype of PACOM

To further verify the communication performance of PACOM, we develop the prototype equipped with complete functions based on the BitTorrent module [50]. The basic configuration is consistent with the simulation setup in Section 6.1. Stegonagraphy is used to boostrap new coming clients to join the network. Furthermore, as discussed in Section 5, PACOM clients in the Private Use Case use stegonagraphy to embed data into file blocks, while the ones in the Public Use Case construct blocks as encrypted data segments with the same size as normal BitTorrent file blocks.

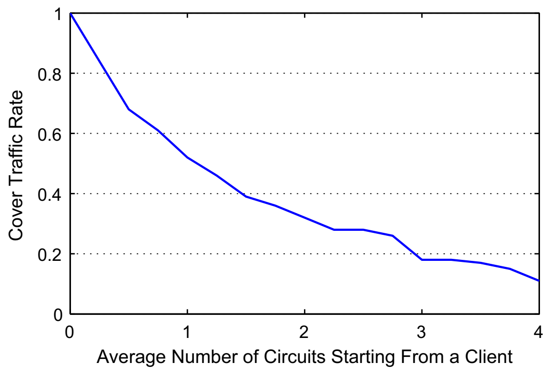

**Fig. 18.** Average cover traffic rate of a PACOM client.

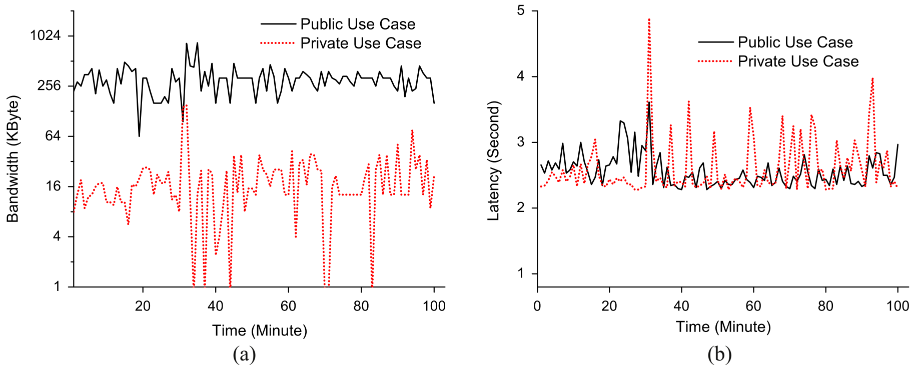

**Fig. 19.** The efficiency to transfer messages between two PACOM clients. (a) The bandwidth of transferring messages. (b) The end-to-end latency to send a message from a PACOM client to another one.

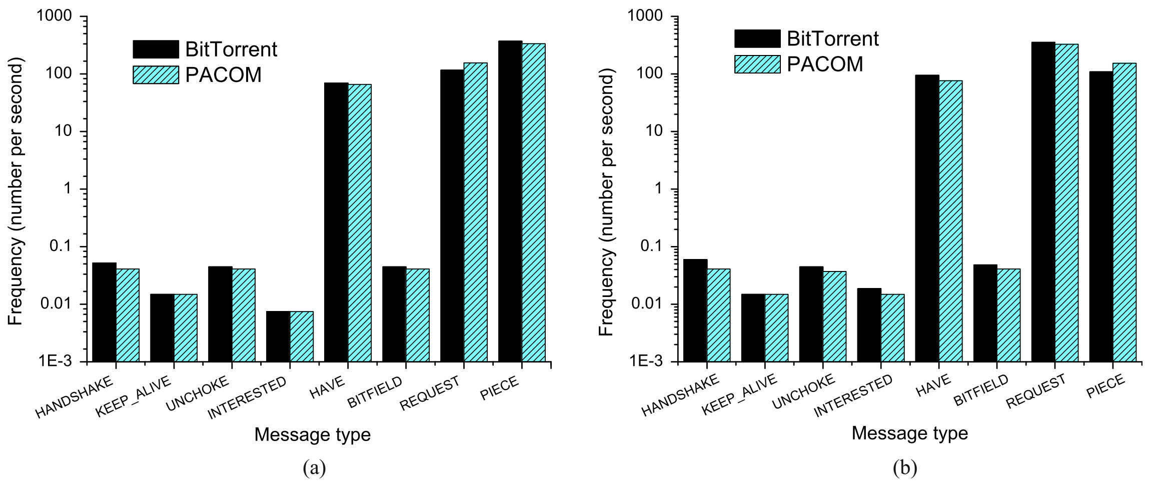

**Fig. 20.** Frequency of different types of BitTorrent messages. (a) Incoming BitTorrent messages. (b) Outgoing BitTorrent messages.

We deploy 100 clients distributed in our campus to test the performance in both use cases. Each client randomly selects another one as its partner to perform end-to-end communication. Fig. 19 shows the result of the communication between one randomly selected client and its partner. As illustrated in Fig. 19(a), the bandwidth of communication in the Private Use Case is about 21 kB/s on average, while the bandwidth in the Public Use Case is about 314 kB/s on average. Moreover, Fig. 19(b) indicates that the end-to-end latency in both cases is close to 2.5 s.

The above experimental results show that PACOM in the Private Use Case is competent to transfer messages or files of small size. Meanwhile, PACOM in the Public Use Case is efficient to support various kinds of file sharing, chatting, or accessing some web based services such as cloud storage and LBS. The high efficiency of PACOM can be due to the special packet scheduling mechanism compatible with BitTorrent. Each PACOM node performs like a BitTorrent client and sends out file blocks embedding communication data when receiving block requests. According to the BitTorrent protocol, the block requests among BitTorrent clients are frequent and a large amount of traffic is generated to transfer file blocks, so PACOM nodes can efficiently send out messages encapsulated in file blocks.

To verify the theoretical analysis about the effectiveness against protocol testing attacks in Section 4.2, we also compare the statistical traffic properties of the PACOM client and the BitTorrent client [50]. Specifically, Fig. 20 illustrates the frequency of different types of incoming and outgoing BitTorrent messages, which are defined in the BitTorrent protocol [45]. Fig. 21 shows the cumulative distribution function of timing interval between transferred BitTorrent messages and Fig. 22 shows the distribution of message size. All of above experimental results indicate that the statistical traffic pattern of the PACOM client is very close to that of a normal BitTorrent client.

## 7. Related work

Since Chaum [15] proposed the first mix-based anonymous communication system in 1981, a lot of anonymous communication systems have been presented. These sys-

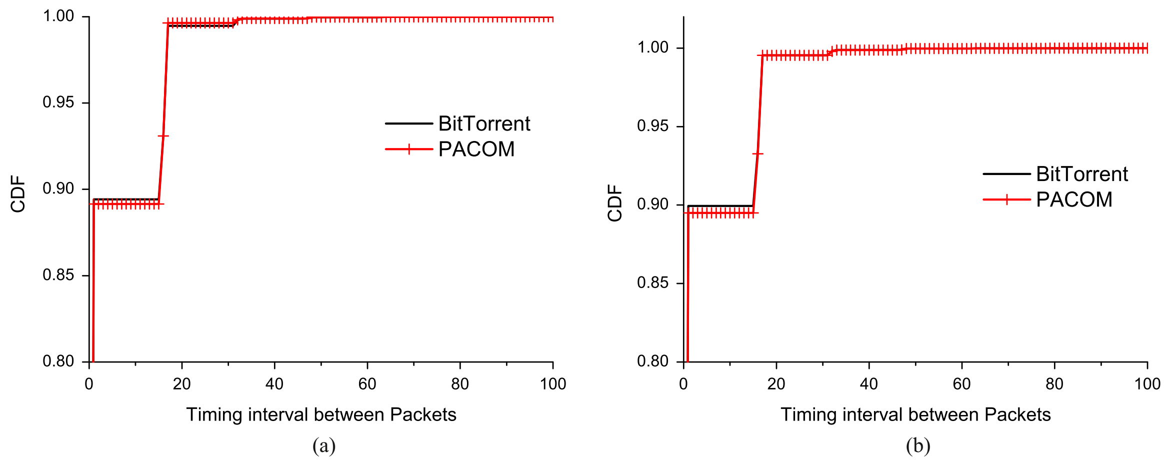

**Fig. 21.** Cumulative distribution function of timing interval between BitTorrent messages. (a) Timing interval between incoming BitTorrent messages. (b) Timing interval between outgoing BitTorrent messages.

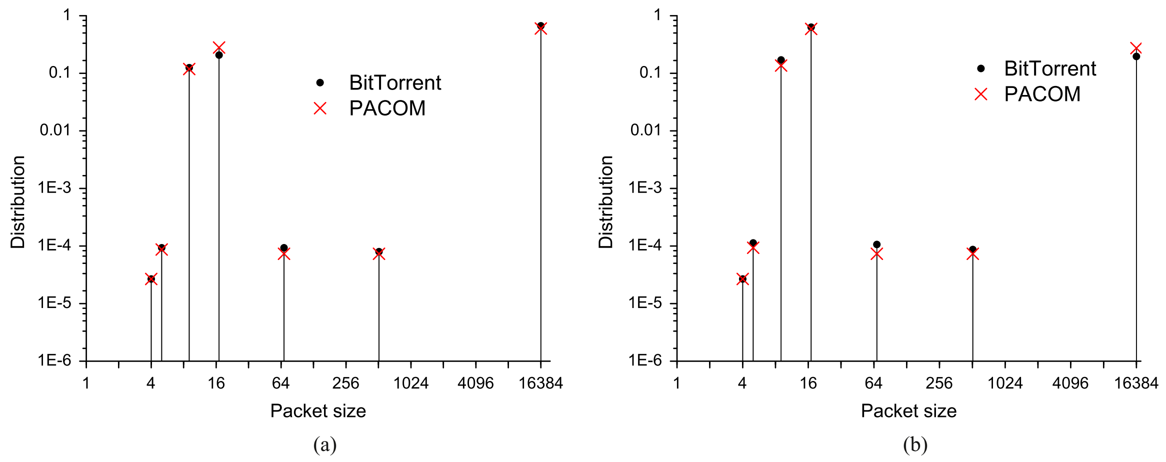

**Fig. 22.** Distribution of the size of transferred BitTorrent messages. (a) Size of incoming BitTorrent messages. (b) Size of outgoing BitTorrent messages.

tems can be classified into two categories: high-latency and low-latency systems. The high-latency systems such as Mixminion [16] and Mixmaster [27] are designed to preserve the data anonymity against the powerful global attackers, but the communication latency can be more than several hours.

The low-latency anonymous systems can be further divided into core-mix-network based and Peer-to-Peer network based systems [21]. In the core-mix-network based systems including OnionRouting [1], Freedom[28], and Tor[2], all clients build the circuits with the help of some public mix servers to conduct anonymous communication. The scalability of the mix network can be a serious problem in these systems when the number of clients increases sharply. In the Peer-to-Peer network based systems, each client in the network is also a mix. The communication circuits can pass through any client in the network. Crowds [3], Tarzan [4], MorphMix [5], ShadowWalker [11], AP3 [6], Salsa [10], NISAN [7], Torsk [8], and Rumor Riding [9] belong to this category. Most of the above systems exhibit a common characteristic: the traffic patterns of adjacent links in any anonymous communication path are statistically correlated. This is a potential entry for the traffic analysis attacks to deduce the communication path successfully.

A lot of attacks [14,21–24,29–38] have been put forward to undermine the anonymity of the low-latency anonymous systems. In particular, the traffic analysis attacks discussed in this paper provide powerful tools to attack a wide spectrum of mix networks. The attacks are conducted to observe the traffic pattern to correlate the links on the same communication path. According to the strategies to obtain the traffic pattern, these attacks can be classified into two categories roughly: passive attacks [14,21,31–34] and active attacks [22–24,30].

In the passive attacks, the adversaries only observe the packet flows and analyse the timing pattern to determine the correlation among the flows. Serjantov and Sewell

[32] correlate the incoming and outgoing links of a node by analyzing the similarity of packet counting and increase of traffic volume. Øverlier and Syverson [31] combine the packet counting analysis [32] and the predecessor attack [29] to locate the hidden servers in Tor [2]. Levine et al. [14] investigate the traffic analysis attacks by packet counting and suggest the defensive dropping of packets to thwart the attack. Danezis [33] and Troncoso and Danezis [34] present a probability model of the mix networks and analyse the probability of the traffic in different links belonging to the same communication path. Zhu et al. [21] propose a passive attack based on the traffic analysis in the frequency domain, and prove that the attack remains effective even if some additional batching strategies are adopted in the mix nodes.

In the active attacks [22–24,30], the adversaries try to transform the attacked packet flow to observe how the traffic of other packet flows gets affected and build the correlation among the flows. Wang et al. [22] encode a watermark into the attacked packet flow by tuning the time intervals of packets, and then try to decode the watermark in other flows to find the correlated ones. The research [22] points out the traditional flow transformation strategies, such as traffic padding [1,2], cover traffic adding [4,12,13], packet dropping [14], flow mixing [15–17] and batching [16,18], all fail to resist this kind of attacks. Murdoch and Danezis [23] modulate the traffic of the corrupted target server, and judge whether a mix node is on the communication path by measuring the variation of its load. In a similar way, Chakravarty et al. [24] inject a number of short or one large burst of traffic into the colluding server and identify the possible anonymous receiver by testing the change of available bandwidth in the edge network. Ling et al. [30] manipulate the Tor cells sent from a sender to cause some decryption errors, which are recognized at the exit onion router to correlate the two ends of a Tor circuit.

Recently, some special defence strategies [19,20,39,40] against the traffic analysis attacks were proposed. Wang et al. [19] put forward the Dependent Padding Algorithm (DLP) to make all outgoing links of a node have exactly the same packet timing which is a matched schedule for all incoming links. Wright et al. [20] morph traffic in each link to look like another different protocol by using convex optimization techniques to transform the packets size distribution. The above methods are effective against the passive attacks, but cannot prevent the active attacks [23,24]. While the active adversaries modulate the communication traffic of a colluding client, some change of the traffic pattern will emerge in the links of the communication path containing the client. Feigenbaum et al. [39] propose the timestamp based technique to prevent active attacks, which forwards the message at predetermined time. This method needs to synchronize the time of each intermediary router and client, thus making the system inflexible and easily broken by malicious routers. Kim et al. present RAD [40], which adopts some public routing proxies to multicast messages to a set of $k$ network entities including the intended message receiver. RAD is effective to resist the traffic analysis by mixing the receiver with other multicast group members, but it is not cost-efficient to achieve large anonymity set by expanding the multicast group.

Beyond traditional mix networks, some circumvention systems such as Skype-Morph [51], CensorSpoofer [52] and StegoTorus [53] are recently proposed to hide communication by imitating popular protocol such as HTTP, Skeype video calls, and Sip based Voice-over-IP. The major differences between PACOM and these systems are summarized as follows:

(1) The systems [51–53] only mimic some target protocols such as Skeype, but not perform the real protocols. They transform the timing interval, packet size and message format of transferred data based on the collected historical trace of target protocol. However, the research [54] argues that these systems fail to mimic all detailed issues of the protocol such as reaction to errors, user behaviors, and other implementation-specific artifacts. This makes mimic traffic easy to be distinguished from the genuine one. The research [54] also points out that one promising alternative is to not mimic, but run the actual protocol and hide the data in the genuine flows. This is consistent with the principle of PACOM. As illustrated in Fig. 2, each PACOM client runs the BitTorrent module [50] and joins the BitTorrent network. Communication data is embedded into transferred file blocks among PACOM clients. The traffic of each link is triggered by the BitTorrent protocol. This is not mimic, but running the real BitTorrent protocol.

(2) Their design goals are different. The systems [51–53] are designed for censorship circumvention and transforming the flows between two directly connected nodes. They do not provide anonymity protection by themselves. A typical use case is to be combined with some anonymous communication systems such as Tor [2] to strengthen their circumvention ability. Unlike these systems, PACOM is designed to provide complete anonymous communication functions and resist traffic analysis attacks effectively.

## 8. Conclusion

In this paper, we propose PACOM, a novel parasitic anonymous communication system immune to the traffic analysis attacks that can effectively attack the state-of-the-art anonymous systems. The PACOM clients are parasitic in the BitTorrent network and hide the anonymous communication in the file transferring traffic compatible with the BitTorrent protocol. The traffic patterns of the links belonging to the same PACOM circuit are independent, which prevents the adversaries from correlating the links via traffic analysis. The mix of the PACOM network and the BitTorrent network containing millions of clients in the Internet also enlarges the effective anonymity set size of PACOM enormously. Both theoretical analysis and comprehensive experiments show that the PACOM network is scalable, effective, and efficient in resisting the traffic analysis attacks.

## Acknowledgements

The work described in this paper was supported by the grants from National Natural Science Foundation of China (Nos. 61300221, 61471217), the Fundamental Research Funds for the Central Universities (No. 2014ZZ0038), the Comprehensive Strategic Cooperation Project of Guangdong Province and Chinese Academy of Sciences (No. 2012B09 0400016), the Technology Planning Project of Guangdong Province (No. 2012A011100005), the High-Tech Research and Development Program of China (863 – "China Cloud" Major Program, No. SQ2015AAJY1595), and the China Postdoctoral Science Fund (No. 2014M550735).

## Appendix A. Steganography mechanism used in boostrapping

The steganography mechanism adopted in the boostrapping procedure is to hide an integer ID $x$ in the BitTorrent traffic. We choose the BitTorrent *HAVE* message to hide information, which is broadcast from a BitTorrent client to all of its neighbors when finishing downloading a file block. The message body contains the ID of the finished file block. By collecting the *HAVE* messages sent from a client, its neighbor can achieve a sequence of IDs, which is named as *ID-sequence* in this paper. Any client's *ID-sequence* is determined by the download order of file blocks in this client. Thus a client can hide information by tuning the download order of file blocks to make change of its *ID-sequence*, and its neighbors can decode the information by observing how the *ID-sequence* changes.

Algorithm 2 presents the procedure of a boostrapping node to decide which file block to download when it plans to hide the ID $x$ in its *ID-sequence*. In the algorithm, the missing file blocks of the client are divided into two classes by using $H_x$, which is the No. $x$ function in a set of uniform hashing functions $\{H_i | 1 \leqslant i \leqslant 100\}$. The client does not select the blocks in the class 2 to download unless all the blocks in the class 1 have been downloaded before. In this way, its *ID-sequence* may appear to follow a strong clustering pattern, and the hidden ID can be decoded by the client knowing the hashing functions. Algorithm 3 shows how a client decodes the hidden ID from the collected *ID-sequence* received from another client.

On the other hand, from the view of a adversary who does not know the hashing functions $\{H_i\}$, the *ID-sequence* of a boostrapping node follows a randomly distributed pattern like BitTorrent clients. Thus the adversaries under Threat Model I cannot decode the information and identify the boostrapping nodes, which have same traffic pattern as normal BitTorrent clients.

**Algorithm 2.** A boostrapping node decides which file block to download, when it plans to hide the ID $x$ in its *ID-sequence*.

```
Input: $P_b$: A boostrapping node. $x$: The integer to be
    hidden ($1 \leqslant x \leqslant 100$).
Output: $b$: The file block selected by $P_b$ to download.
Notations:
$m$: The total number of blocks of the downloading file.
$H_i$ ($1 \leqslant i \leqslant 100$): a uniform hash function mapping
    the integers in $[0, m-1]$ to the ones in the same
    scope.
$B(P_b)$: The collection of file blocks owned by $P_b$.
$b_k$: A file block.
$C1$: The set of file blocks in class 1.
$C2$: The set of file blocks in class 2.
Method:
1 for $\forall b_k \notin B(P_b)$ do
2   if $H_x(b_k) < m/2$ then
3     $C1 = C1 \cup \{b_k\}$
4   else
5     $C2 = C2 \cup \{b_k\}$
6 if $|C1| > 0$ then
7   $b \leftarrow$ a block randomly selected from C1
8 else
9   $b \leftarrow$ a block randomly selected from C2
```

**Algorithm 3.** A PACOM client decodes an integer ID from the collected *ID-sequence* received from another one.

```
Input:
$P_i$: A PACOM client.
$L_j$: The ID-sequence received from another client $P_j$
$n$: The size of $L_j$.
$L_{jk}$: The No. k integer in the $L_j$.
Output: The decoded integer ID is returned. If the
    decoding fails, $-1$ is returned.
Notations:
$m$: The total number of the blocks of the downloading
    file.
$H_i$ ($1 \leqslant i \leqslant 100$): a uniform hash function mapping
    the integers in $[0, m-1]$ to the ones in the same
    scope.
$c$: An constant integer indicating the size of decoding
    scope.
Method:
1 for $i \leftarrow 1$ to 100 do
2   $y \leftarrow 0$
3   //Try to decode the integer ID from the last $c$
    elements of $L_j$.
4   for $k \leftarrow n-c+1$ to $n$ do
5     if $H_i(L_{jk}) < m/2$ then
6       $y \leftarrow y+1$
7   if $y = c$ or $y = 0$ then
8     return i
9 return $-1$
```

## Appendix B. Proof of Theorem 2

**Proof.** A client is called to be in the *trap-mode* when the PACOM swarms containing the client are all malicious swarms. When observing that a client in the trap-mode sends out a message but has not received a message before, adversaries can judge it as a information sender. However, for a client not in the trap-mode, adversaries cannot make such a decision and the probability for the client to be a communication participant is equal to that of any PACOM client else.

Because the ratio of malicious boostrapping node is $\rho$, the probability of a client contacting a malicious boostrapping node to join a malicious swarm is also $\rho$. Thus the probability of the client in trap-mode is $\rho^r$ and totally there are $n\rho^r$ clients are in trap-mode in the network on average. Each client in trap-mode can be determined to be a information sender or not, so the entropy is calculated as follows.

$$-\sum_{P_u \in T} Pr(P_u)\log(Pr(P_u)) = 0 \tag{B.1}$$

Here $T$ means the set of clients in trap-mode. For the rest of $n(1-\rho^r)$ clients not in trap-mode, the probability for each one to be the information sender is equal to $1/(n(1-\rho^r))$. Thus the effective anonymity set size of PACOM can be calculated as follows:

$$\begin{aligned} S &= -\sum_{P_u \in T} Pr(P_u)\log(Pr(P_u)) - \sum_{P_u \notin T} Pr(P_u)\log(Pr(P_u)) \\ &= 0 - \log(1/(n(1-\rho^r))) = \log(n(1-\rho^r)) \qquad \square \end{aligned}$$

## References

[1] D. Goldschlag, M. Reed, P. Syverson, Onion routing, Commun. ACM 42 (2) (1999) 39–41.

[2] R. Dingledine, N. Mathewson, P. Syverson, Tor: the second generation onion router, in: Proc. 13th USENIX Security Symposium, vol. 13, 2004.

[3] M. Reiter, A. Rubin, Crowds: anonymity for web transactions, ACM Trans. Inform. Syst. Sec. 1 (1) (1998) 66–92.

[4] M. Freedman, R. Morris, Tarzan: a peer-to-peer anonymizing network layer, in: Proc. 9th ACM Conf. Computer and Communications Security (CCS02), 2002, pp. 193–206.

[5] M. Rennhard, B. Plattner, Introducing morphmix: peer-to-peer based anonymous Internet usage with collusion detection, in: Proc. ACM Workshop on Privacy in the Electronic Society (WPES'02), 2002, pp. 91–102.

[6] A. Mislove, G. Oberoi, A. Post, C. Reis, P. Druschel, D. Wallach, Ap3: cooperative, decentralized anonymous communication, in: Proc. ACM SIGOPS European Workshop, 2004.

[7] A. Panchenko, S. Richter, A. Rache, Nisan: network information service for anonymization networks, in: Proc. 16th ACM Conf. Computer and Communications Security (CCS09), 2009, pp.141–150.

[8] J. McLachlan, A. Tran, N. Hopper, Y. Kim, Scalable onion routing with torsk, in: Proc. 16th ACM Conf. Computer and Communications Security (CCS09), 2009, pp. 590–599.

[9] Y. Liu, J. Han, J. Wang, Rumor riding: anonymizing unstructured peer-to-peer systems, IEEE Trans. Parallel Distrib. Syst. 22 (3) (2011) 464–475.

[10] A. Nambiar, M. Wright, Salsa: a structured approach to large-scale anonymity, in: Proc. 13th ACM Conf. Computer and Communications Security (CCS06), 2006, pp. 17–26.

[11] P. Mittal, N. Borisov, Shadowwalker: peer-to-peer anonymous communication using redundant structured topologies, in: Proc. 16th ACM Conf. Computer and Communications Security (CCS09), 2009, pp. 161–172.

[12] Y. Guan, X. Fu, D. Xuan, P. Shenoy, R. Bettati, W. Zhao, Netcamo: camouflaging network traffic for Qos guaranteed mission critical applications, IEEE Trans. Syst., Man, Cybernet. 31 (4) (2001) 253–265.

[13] Y. Guan, C. Li, D. Xuan, R. Bettati, W. Zhao, Preventing traffic analysis for real-time communication networks, in: Proc. IEEE Military Communications Conference (MIL-COM99), 1999, pp. 744–750.

[14] B. Levine, M. Reiter, C. Wang, M. Wright, Timing attacks in low-latency mix systems, in: Proc. 8th Intl Conf. Financial Cryptography, 2004, pp. 251–265.

[15] D. Chaum, Untraceable electronic mail, return addresses, and digital pseudonyms, Commun. ACM 24 (2) (1981) 84–90.

[16] G. Danezis, R. Dingledine, N. Mathewson, Maxminion: design of a type III anonymous remailer protocol, in: Proc. IEEE Symp. Security and Privacy, 2003, pp. 2–15.

[17] A. Jerichow, J. Muller, A. Ptzmann, B. Ptzmann, M. Waidner, Real-time MIXes: a bandwidth-effcient anonymity protocol, IEEE J. Sel. Areas Commun. 16 (4) (1998) 495–509.

[18] A. Serjantov, R. Dingledine, P. Syverson, From a trickle to a flood: active attacks on several mix types, in: Proc. Information Hiding Workshop (IH 02), 2002, pp. 36–52.

[19] W. Wang, M. Motani, V. Srinivasan, Dependent link padding algorithms for low latency anonymity systems, in: Proc. 15th ACM Conference on Computer and Communications Security (CCS08), 2008, pp. 323–332.

[20] C. Wright, S. Coull, F. Monrose, Traffic morphing: an efficient defense against statistical traffic analysis, in: Proc. 16th Network and Distributed Security Symposium (NDSS'09), 2009, pp. 237–250.

[21] Y. Zhu, X. Fu, B. Graham, R. Bettati, W. Zhao, Correlation-based traffic analysis attacks on anonymity networks, IEEE Trans. Parallel Distrib. Syst. 21 (7) (2010) 954–967.

[22] X. Wang, S. Chen, S. Jajodia, Network flow watermarking attack on low-latency anonymous communication systems, in: Proc. IEEE Symp. Security and Privacy, 2007, pp. 116–130.

[23] S.J. Murdoch, G. Danezis, Low-cost traffic analysis of Tor, in: Proc. IEEE Symp. Security and Privacy, 2005, pp. 183–195.

[24] S. Chakravarty, A. Stavrou, A. Keromytis, Traffic analysis against low-latency anonymity networks using available bandwidth estimation, in: Proc. 15th European Symp. Research in Computer Security (ESORICS10), 2010, pp. 249–267.

[25] BitTorrent Inc, uTorrent & BitTorrent Surge to 150 Million Monthly Users, 2012. <http://torrentfreak.com/bittorrent-surges-to-150-million-monthly-users-120109/>.

[26] Andrei Serjantov, George Danezis, Towards an information theoretic metric for anonymity, in: Proc. 2th Privacy Enhancing Technologies Symposium (PET02), vol. 2482, 2002, pp. 41–53.

[27] U. Moller, L. Cottrell, P. Palfrader, L. Sassaman, Mixmaster Protocol Version 2, IETF Internet Draft, 2003.

[28] A. Back, I. Goldberg, A. Shostack, Freedom Systems 2.1 Security Issues and Analysis, White Paper, Zero Knowledge Systems, Inc., 2001.

[29] M. Wright, M. Adler, B. Levine, C. Shields, The predecessor attack: an analysis of a threat to anonymous communications systems, ACM Trans. Inform. Syst. Sec. 7 (4) (2004) 489–522.

[30] Z. Ling, J. Luo, W. Yu, X. Fu, W. Jia, W. Zhao, Protocol-level attacks against Tor, Comput. Networks 57 (4) (2013) 869–886.

[31] L. Øverlier, P. Syverson, Locating hidden servers, in: Proc. IEEE Symp. Security and Privacy, 2006, pp. 100–114.

[32] A. Serjantov, P. Sewell, Passive attack analysis for connection-based anonymity systems, in: Proc. 8th European Symp. Research in Computer Security (ESORICS03), 2003, pp. 116–131.

[33] G. Danezis, The traffic analysis of continuous-time mixes, in: Proc. Privacy Enhancing Technologies Workshop (PET04), vol. 3424, 2004, pp. 35–50.

[34] C. Troncoso, G. Danezis, The Bayesian traffic analysis of mix networks, in: Proc. 16th ACM Conf. Computer and Communications Security (CCS09), 2009, pp. 369–379.

[35] D. Kesdogan, D. Agrawal, S. Penz, Limits of anonymity in open environments, in: Proc. Information Hiding Workshop (IH02), vol. 2578, 2002, pp. 53–69.

[36] G. Danezis, Statistical disclosure attacks: traffic confirmation in open environments, in: Proc. Security and Privacy in the Age of Uncertainty (SEC03), 2003, pp. 421–426.

[37] D. Kesdogan, D. Agrawal, V. Pham, D. Rautenbach, Fundamental limits on the anonymity provided by the MIX technique, in: Proc. IEEE Symp. Security and Privacy, 2006, pp. 86–99.

[38] Q. Wang, P. Mittal, N. Borisov, In search of an anonymous and secure lookup, in: Proc. 17th ACM Conf. Computer and Communication Security (CCS 10), 2010, pp. 308–318.

[39] J. Feigenbaum, A. Johnson, P. Syverson, Preventing active timing attacks in low-latency anonymous communication, in: Proc. 10th Privacy Enhancing Technologies Symposium (PETS10), 2010, pp. 166–183.
[40] H. Kim, J. Jeong, RAD: recipient-anonymous data delivery based on public routing proxies, Comput. Networks 55 (15) (2011) 3469–3484.
[41] Wikipedia, Comparison of BitTorrent Clients, 2012. <http://en.wikipedia.org/wiki/Comparison_of_BitTorrent_clients>.
[42] A. Loewenstern, DHT Protocol, 2008. <http://www.bittorrent.org/beps/bep_0005.html>.
[43] P. Maymounkov, D. Mazieres, Kademlia: a peer-to-peer information system based on the XOR metric, The Intl Workshop on Peer-to-Peer Systems, 2002, pp. 53–65.
[44] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, X. Zhang, Measurements, analysis, and modeling of BitTorrent-like systems, in: Proc. Internet Measurement Conf., 2005, pp. 35–48.
[45] Theory.org, Bittorrent Protocol Specification v1.0, 2006. <http://wiki.theory.org/BitTorrentSpecification>.
[46] F. Petitcolas, R. Anderson, M. Kuhn, Information hiding: a survey, in: Proc. of the IEEE, Special Issue on Protectionon Multimedia Content, vol. 87, no. 7, 1999, pp. 1062–1078.
[47] W. Diffie, M.E. Hellman, New directions in cryptography, IEEE Trans. Inform. Theory 22 (6) (1976) 644–654.
[48] T. Hoßfeld, F. Lehrieder, D. Hock, S. Oechsner, Z. Despotovic, W. Kellerer, M. Michel, Characterization of BitTorrent swarms and their distribution in the Internet, Comput. Networks 55 (5) (2011) 1197–1215.
[49] TATA Consultancy Services, WANem: The Wide Area Network Emulator. <http://wanem.sourceforge.net/>.
[50] Java Bittorrent API. <http://bitext.sourceforge.net/>.
[51] H. Moghaddam, B. Li, M. Derakhshani, I. Goldberg, SkypeMorph: Protocol obfuscation for Tor bridges, in: CCS, 2012.
[52] Q. Wang, X. Gong, G. Nguyen, A. Houmansadr, N. Borisov, CensorSpoofer: asymmetric communication using IP spoofing for censorship-resistant web browsing, in: CCS, 2012.
[53] Z. Weinberg, J. Wang, V. Yegneswaran, L. Briesemeister, S. Cheung, F. Wang, D. Boneh, StegoTorus: a camouflage proxy for the Tor anonymity system, in: CCS, 2012.
[54] A. Houmansadr, C. Brubaker, V. Shmatikov. The parrot is dead: observing unobservable network communications, in: CCS, 2013.
[55] G. Danezis, C. Troncoso, Vida: how to use Bayesian inference to de-anonymize persistent communications, in: Privacy Enhancing Technologies, Springer, Berlin Heidelberg, 2009, pp. 56–72.
[56] F. Prez-Gonzlez, Fernando, C. Troncoso, Understanding statistical disclosure: a least squares approach, in: Privacy Enhancing Technologies, Springer, Berlin Heidelberg, 2012, pp. 38–57.
[57] J. Raymond, Traffic analysis: protocols, attacks, design issues, and open problems, in: Workshop on Designing Privacy Enhancing Technologies, 2001.
[58] M. Wright, M. Adler, B. Levine, C. Shields, Passive-logging attacks against anonymous communications systems, ACM Trans. Inform. Syst. Sec. 11 (2) (2008) 1062–1078.
[59] A. Bellissimo, B. Levine, P. Shenoy, Exploring the Use of BitTorrent as the Basis for a Large Trace Repository, Tech. Rep. 04-41, University of Massachusetts Amherst, June 2004.
[60] UMass Trace Repository, May 17, 2014. <http://traces.cs.umass.edu/index.php/Network/Network>.

**Jianming Lv** received the BS degree in Computer Science from Sun YAT-SEN University, China, in 2002, and the PhD degrees from Institute of Computing Technology, Chinese Academy of Sciences University in 2008. He is currently a lecturer in South China University of Technology. His research interests include distributed computing, peer-to-peer networks, security and privacy. He is a member of IEEE, ACM and CCF.

**Tieying Zhang** is currently an assistant professor at the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include computer networks, distributed computing, peer-to-peer systems, multimedia networking, and network security. He has published over 20 technical papers and book chapters in the above areas. He is a member of IEEE and ACM.

**Zhenhua Li** is a Researcher at the TNLIST (Tsinghua National Lab of Information Science and Technology), and a PostDoc at the School of EECS, Tsinghua University, Beijing, China. He was a joint Ph.D. student at Peking University, China and the University of Minnesota - Twin Cities, USA. His current research areas mainly consist of cloud computing/storage, Internet content distribution, and peer-to-peer technologies. He is a member of ACM, IEEE and CCF.

**Xueqi Cheng** is a professor in Institute of Computing Technology, Chinese Academy of Sciences. He is also a Ph.D. advisor, deputy chief engineer, Director of Research Center of Web Data Science & Engineering. Apart from these, He has become an honorary Professor in Northeastern University and a visiting Ph.D. advisor in China University of Technology. His current research interests include Network Science, Internet information security, Web Search and Mining, Complex Network and Social Computing. He has published more than 200 technical papers in the above areas. He is a member of IEEE and Senior Member of CCF.