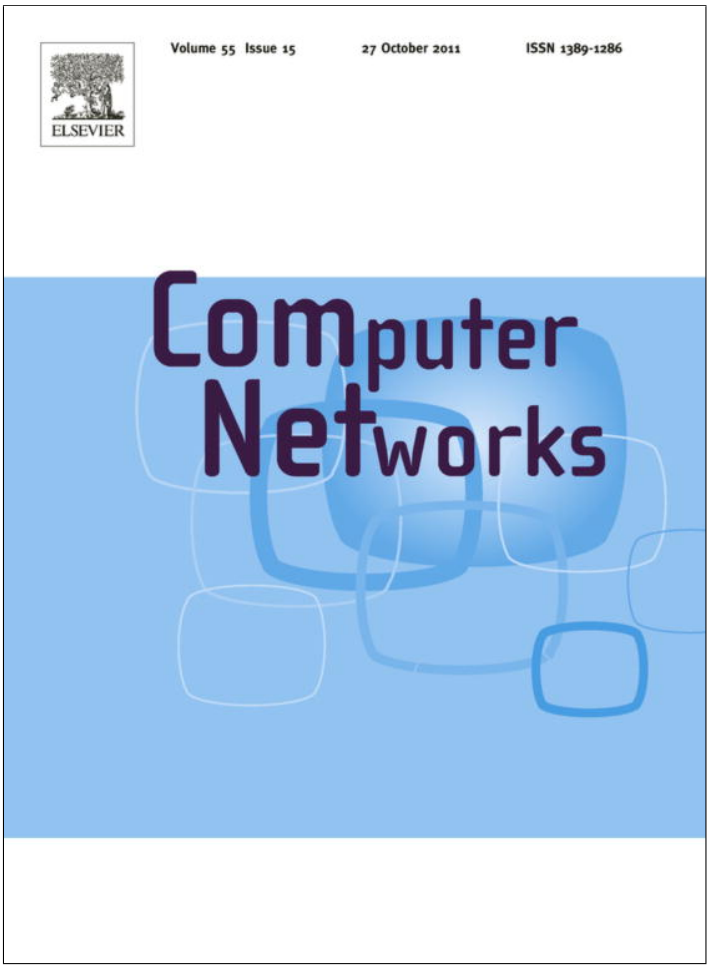
Provided for non-commercial research and education use.
Not for reproduction, distribution or commercial use.



This article appeared in a journal published by Elsevier. The attached copy is furnished to the author for internal non-commercial research and education use, including for instruction at the authors institution and sharing with colleagues.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

http://www.elsevier.com/copyright

Author's personal copy

Computer Networks 55 (2011) 3469–3484

Contents lists available at ScienceDirect

Computer Networks

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

RAD: Recipient-anonymous data delivery based on public routing proxies

Hyoungshick Kim [a,*], Jaehoon Jeong [b]

[a] Computer Laboratory, University of Cambridge, United Kingdom
[b] Department of Computer Science and Engineering, The University of Minnesota, United States

ARTICLE INFO

Article history:
Received 14 September 2010
Received in revised form 31 May 2011
Accepted 7 July 2011
Available online 22 July 2011

Keywords:
Anonymous communication
Traffic analysis
Multicast tree
Public routing proxy

ABSTRACT

This paper presents a *Recipient-Anonymous Data Delivery* (*RAD*), tailored and optimized for stable-topology networks. There is one simplistic approach for achieving *recipient anonymity*. If a message is equally broadcasted to all network entities (e.g., routers and hosts), an adversary cannot infer any traffic patterns for the intended recipient.

While this technique is unconditionally secure, no one believes that this is a practical solution except in some special network environments since it requires extremely expensive traffic overhead. In this paper we realize this idea with an efficient multicast protocol by introducing the concept of a *public routing proxy*. A *public routing proxy* enables a sender to deliver a message to the intended recipient anonymously since the *public routing proxy* multicasts the message to a set of k network entities including the intended recipient. Thus, in the proposed protocol, the recipient's k-anonymity can be unconditionally guaranteed. We also demonstrate the practicality of the proposed protocol through intensive simulation based on well-known network topologies.

© 2011 Elsevier B.V. All rights reserved.

1. Introduction

Using traffic analysis, it is possible to infer who communicates with whom over open networks. In conventional networks, a network entity should inherently identify its correspondent to communicate with. For example, in packet switched networks, the network-level addresses to identify the source and destination are visible to anyone with access to any link over which the traffic flows. Since the address information is commonly contained in not the packet contents but the packet headers, traffic analysis is still effective even if the packet content is encrypted. Therefore traffic analysis is relatively easy compared to cryptanalysis that requires considerable efforts to break [7]. In many applications, traffic analysis is a key threat to the right to user privacy. For example, when a customer placing an online order wants to make his/her transactions anonymous, an attacker can prevent it by tracing the route of the network packets. In this paper, we propose an efficient, recipient-anonymous network protocol to be resistant to traffic analysis in stable-topology networks such as the Internet.

Many network protocols against traffic analysis have been developed. These protocols should be efficient while hiding the information about the communicating parties. For example, theoretically secure *recipient anonymity* can be simply achieved by using broadcast [13] if we do not care efficiency of the protocol. In other words, a sender broadcasts a message to all network entities including the intended recipient. For this approach, even an adversary with unlimited computing power cannot infer any information about the intended recipient since a message is delivered to every recipient in the same manner. However, this technique may be impractical due to its inherent huge traffic overhead. Therefore, most research efforts have elaborated on reasonable trade-offs between privacy and efficiency. The most popular approach [10,27] is to obfuscate the traffic route with cryptographic primitives such as public key cryptography. In theory, public key cryptography may provide a neat solution for anonymous

* Corresponding author.
E-mail addresses: hk331@cl.cam.ac.uk (H. Kim), pjeong@brocade.com (J. Jeong).

1389-1286/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2011.07.009

Author's personal copy

communication. In practice, however, public key infrastructure (PKI) has proved to be difficult to deploy [12]; it is expensive and the existence of trusted parties is required. Above all, the most serious problem of these solutions is not effective against *correlation attacks* [18] where attackers analyse the information about traffic entering and exiting (e.g. timing information) and determine with whom one is communicating using correlation functions.

Alternatively, we focus on designing an anonymous protocol without use of any cryptographic operations to hide the relationship between incoming and outgoing messages against not only *traffic tracing attacks* but also *correlation attacks*. This paper particularly investigates *one-way k-recipient-anonymous* data delivery services, such as privacy-sensitive health-related advertisement and military operation command delivery. The main purpose is to blur who is the actually intended message receiver from others by generating faked messages.

We generalize the problem as follows: Alice wants to deliver a secret letter to Bob. Alice does not want to reveal her original recipient. Eve can observe all messages between any users over the network. How can Alice hide her original recipient from Eve when Alice tries to deliver the secret message to Bob? Our main idea is to realize the concept of broadcast for *recipient anonymity* with an efficient multicast-based protocol. The basic process is simply described as follows:

1. Alice chooses a set of network users including Bob, that is called the *anonymity set*.
2. Alice multicasts a message to all members in the *anonymity set* in the same manner.

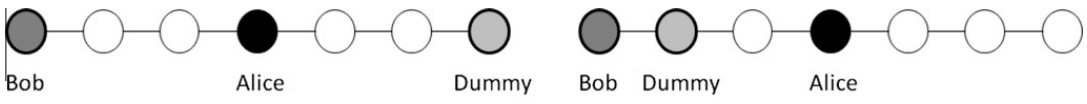
In this protocol, Eve cannot unconditionally identify Bob from the other members in the *anonymity set* [23] since any message traffic pattern between Alice and a member in the *anonymity set* is not specific. This protocol is exactly the same as the broadcast-based scheme when we choose all network users for the *anonymity set*.

However, there are two challenging issues in designing a multicast-based protocol for efficiency. The first problem is how to choose dummy members for *anonymity set*. The efficiency of multicast protocol naturally depends on which users are chosen. Fig. 1 shows the importance of choosing dummy users for *anonymity set*. 6 messages are totally required to deliver a message from Alice to Bob and the dummy user in Fig. 1(a) whereas it is enough to send 3 messages for the same purpose in Fig. 1(b). Therefore we should carefully choose dummy users for *anonymity set*. Unfortunately, finding a set of network users for an efficient multicasting is not trivial.

The second problem is how to construct a *multicast tree* for an *anonymity set*. In general, the problem of multicast routing in communication networks is equivalent to finding a tree $T = (V_T, E_T)$ called a *multicast tree* in a network such that T spans all network users in the multicast group. Note that the performance of multicasting is dramatically changed depending on the constructed tree T. Fig. 2 shows the importance of *multicast tree* optimization. However, this problem is equivalent to a classical optimization problem called the *Steiner tree problem* which is well-known NP-hard [16]. Given an arbitrary weighted graph with a distinguished vertex subset, the *Steiner Tree Problem* asks for a minimum-cost subtree spanning the distinguished vertices [30]. Moreover, the *Steiner tree* is not always an ideal multicast route in all applications. For example, the delay performance can be worse than other multicast routes since it does not constrain the maximum path length [6]. Therefore it is computationally difficult to construct a 'good' *multicast tree* even if the proper *anonymity set* is completely given. This means that we must use heuristics.

However, by introducing the concept of *public routing proxy*, we can support an efficient multicasting for recipient k-anonymity. A proxy is already a well-known concept for anonymous communication. The *mixes* in [5] and the *onion* routers in [27] are the representative examples of proxies for anonymous communication. While they use randomly selected proxies (or routers), we choose a network entity that is topologically close to the recipient, as the recipient's proxy. This paper contributes in the following areas:

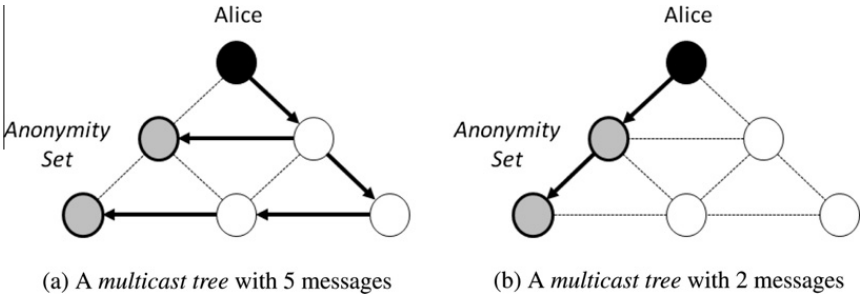
- We introduce the concept of *public routing proxy* considering both of efficiency and anonymity: A sender (Alice) can deliver a message to the intended recipient (Bob) via the recipient's a *public routing proxy*. Our RAD protocol in Section 3 provides perfect k-anonymity without the assumptions of trusted proxies and cryptographic primitives. We formally define the threat model in Section 2 and analyse the security of RAD in Section 5.2.
- We particularly design an algorithm to generate the *public routing proxies* for each network entity in Section 4. We also show that the running time of this algorithm is $O(n^3)$, where n is the number of network entities.
- In Section 5 we empirically analyse the performance and anonymity of RAD with several network datasets in order to show that our RAD protocol can perform well in many network topologies as follows: four random graphs, two Transit-Stub graphs [4], a ring graph, a Watts-Strogatz small world graph [37], a Barabási-Albert scale free network [2], a Content-Addressable Network (CAN) [26], a Chord network [33], and a Hypergrid graph [29]. The experimental results demonstrated that RAD is capable of achieving k-anonymity with low traffic overhead and network latency on these network topologies.



(a) An *anonymity set* with a bad dummy (b) An *anonymity set* with a good dummy

Fig. 1. The importance of choosing dummies for *anonymity set*.

Author's personal copy



(a) A *multicast tree* with 5 messages (b) A *multicast tree* with 2 messages

Fig. 2. The importance of *multicast tree* optimization.

2. Model

2.1. Network model

We view a *network* as an undirected, unweighted and connected graph $G = (V,E)$, where the vertex set V contains the network entities to exchange messages and the edge matrix E represents communication channels between entities. In G, each vertex is identified by a unique identifier.

This paper does not assume any density of data traffic generated in the network. This is different from the assumption of conventional anonymous schemes such as the Chaum's *mix* scheme [5] and *Onion routing* [27]. Unlike conventional anonymous protocols, we do not assume this traffic condition. For *mix* schemes or *Onion routing*, the traffic should be continuously generated in a heavy density so that the messages can be sufficiently shuffled at *mixes* or *onion* routers to hide their relation with the sender. On the other hand, our goal is to provide anonymity regardless of the other network entities' traffic condition.

2.2. Security goals

We assume that a computationally unlimited adversary can try to trace the messages exchanged between network entities; note that the adversary can eavesdrop communication between any two of the participants (i.e., sender, proxy, and recipient) and also control an arbitrary subset of the participants. Moreover, we do not assume the existence of a trusted third party for the roles of administrator or proxy, which is generally required in existing solutions. Since we do not require a trusted proxy, our protocol clearly provides the backward/forward secrecy properties.

For *recipient anonymity*, the adversary should not be able to determine who the intended recipient of a particular message is. The concept of anonymity is defined as the state of not being identifiable with a set of subjects, namely the *anonymity set*. In general, anonymity is defined in terms of *unlinkability* between the item of interest (IOI) and the identifier of a subject. Throughout this paper, we treat "the content of the sender's message" as IOI. In other words, *recipient anonymity* means that "a particular message content (IOI)" is not linkable to "the intended recipient (subject)". Note that "message content" is different from "network transmission". Suppose that Alice wants to deliver a secret message m to Bob. Alice first encrypts m with Bob's public key and then distributes the encrypted message to Bob and $k-1$ other dummies in order to hide that the intended recipient of the message is Bob.

For example, Fig. 3 shows the data delivery to an *anonymity set* where the size of "*anonymity set*" is 5 and an arrow indicates a message transmission. When we treat network transmissions as IOIs, the adversary can identify each network transmission by observing the addressing information attached to each message. However, the intended recipient of the message m can be identified only with the probability of at most 1/5 when we treat "the content of the message m" as an IOI.

Thus, in this paper, our goal is to provide k-anonymity for a recipient. The concept of k-anonymity was originally introduced in the context of relational data privacy [34]. We define that a communication protocol is *recipient k-anonymous* if it can guarantee that an adversary, trying to determine the intended recipient of a particular message, can only narrow down the possible recipients to a set of size k that is named "*anonymity set*" [36]. In particular, we aim to resist not only *traffic tracing attacks* that try to identify the recipient by tracing traffic route but also *correlation attacks* that try to identify two suspected endpoints communicating with each other by using correlation function with traffic entering and exiting.

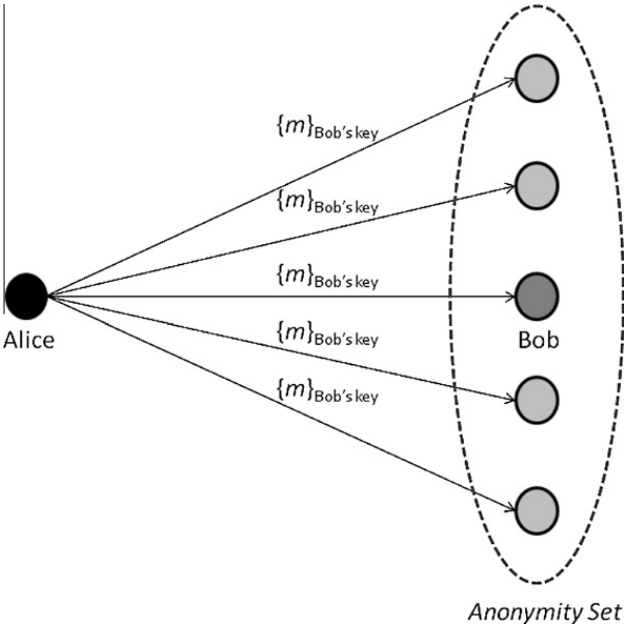


Fig. 3. An example of *recipient anonymity*.

Author's personal copy

3. Anonymous routing protocol

The proposed protocol RAD is basically to distribute the secret message to randomly chosen k-1 dummies and the intended recipient through a *publich routing proxy*; that is, the message of the sender is at first routed to the *public routing proxy* and then the *public routing proxy* delivers the message to the anonymity set of size k. We first define "HopCount" in Definition 3.1 as a basic function used in RAD. Here, $|S|$ and Distance (u, v) denote the cardinality of the set S and the shortest path length between u and v, respectively. Intuitively, HopCount (u,k) can be interpreted as the minimum hop count value needed to satisfy that the number of recipients in multicast is at least k when u delivers a message using the Time-To-Live (TTL)-based multicast.

Definition 3.1. HopCount Given a graph $G = (V,E)$ and a constant k, we define the HopCount for $u \in V$ as:

$$\text{HopCount}(u,k) = \min_{l \in \mathbb{N}} \{ l : |S| \geqslant k \text{ and Distance}(u, v) \leqslant l \text{ for } v \in S \subseteq V \}$$

Now, we explain the RAD protocol with the packet format and the delivery procedure as follows: The RAD protocol packet format is specified in Fig. 4. The field of `Type` is used to identify the "packet type". The fields of `Source Address` and `Destination Address` contain the sender s's network address and the chosen proxy p's network address, respectively. Especially, the field of `Hop Count` contains the information about the TTL value for the multicast initiated by p. With this packet, the data packets will at first be forwarded towards the chosen proxy p and then the proxy p will send the packets to the multicast tree containing the anonymity set. Through this multicast, the packets will be anonymously delivered to the originally intended recipient r. Note that our RAD protocol can be defined as a shim layer between the network layer and the application layer, such as Ad-hoc On Demand Distance Vector Routing (AODV) [22] for Mobile Ad hoc Networking (MANET). Our forwarding protocol consists of the following two steps.

1. **Computing *public routing proxies*.** Given the security parameter k, *public routing proxies* are selected for each vertex $u \in V$ that is a possible message recipient, using Algorithm 1, explained in Section 4. The notation P_u denotes the set of the *public routing proxies* of u. Note that this selection needs to be done only once at the initialization phase if the network topology is not changed. However, this selection requires not only the knowledge about the global network topology but also the considerable computational cost; refer to Section 4 for the detailed discussion. Therefore, in practice, it seems desirable to compute and publish the results by a network entity such as administrator. For example, the administrator can compute *public routing proxies* of all network entities and broadcast them periodically. In fact, anyone with the knowledge about the global network topology can play the role of the administrator. This implies that the *public routing proxies* cannot be forged since the forgery is detectable by anyone with the knowledge about the global network topology. We will show that the average size of *public routing proxies* is reasonable enough; refer to Section 5.1 for the detailed discussion.
   For the scalable computation of Algorithm 1 in a large-scale network, we can divide the whole network into multiple subnetworks and each multiple subnetwork can be maintained by an individual network administrator.
2. **Sending the message.** Before sending the secret message m, the sender s first obtains the intended recipient r's *public routing proxies*, P_r, and then randomly chooses a proxy $p \in P_r$ covering r. The sender s delivers m to p. Upon receiving m,p multicasts m to $v \in M$ where M is the set of vertices within the HopCount (p,k) distance from p. This means that p can forward the message m from s to the vertices in M by simply setting the TTL value as the HopCount (p,k).

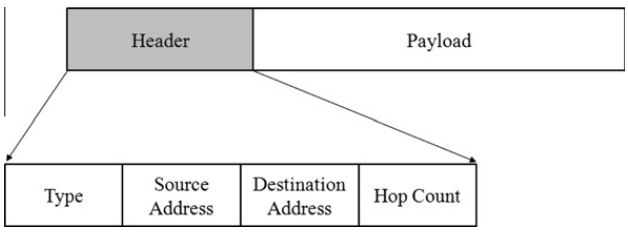


Fig. 4. The proposed packet format.

To give an intuition for our RAD protocol, we consider the scenario depicted in Fig. 5 where the node **A** wants to deliver a message to the node **B** to satisfy that the number of recipients is at least 5 (i.e. 5-anonymity data delivery is required). Before delivering a message, **A** needs to choose one of **B**'s *public routing proxies* randomly. Fig. 5(a) represents the **B**'s five *public routing proxies* (light gray nodes) computed by Algorithm 1. Interestingly, we can see that **B** itself is included in the **B**'s *public routing proxies*. In this scenario, we assume that **A** received the information about all network nodes' *public routing proxies* periodically. Therefore, **A** delivers the message with the information about "`Hop Count =2`" to the randomly chosen node **P** from the **B**'s *public routing proxies*. After receiving this message, the node **P** relays the messages to the nodes within 2 hops. Fig. 5(b) represents the actual recipients (dark gray nodes) in this step.

Note that there is a rule to send multiple messages from the same source to the same recipient sequentially. When a sender wants to deliver messages to the intended recipient successively, the same *public routing proxy* should be used repeatedly. Otherwise, the size of *anonymity set* may be reduced by using statistical inference attacks. For example, suppose that the average number of messages in a communication session is known as 2. In this case, the adversary collects the information about the *public routing proxies*, p and $\bar{p}$, used in delivering two successive messages from the same sender. The adversary can guess that the intended recipient is a vertex in the intersection

Author's personal copy

between the vertices with p and the vertices with $\bar{p}$ as their *public routing proxies*. Therefore, we cannot guarantee the size of this intersection is greater than k.

In Section 1, we discussed two challenging issues: "how to choose dummies" and "how to construct a *multicast tree*" for an *anonymity set*. In RAD, dummies are chosen as the vertices proximate to a *public routing proxy*. These dummies guarantee both of k-anonymity and efficiency. Although finding the minimal *Steiner tree* is NP-hard, the minimal *Steiner tree* is exactly identical to a minimum spanning tree $T = (V_T, E_T)$ of a multicast group M when $M = V_T$. When a graph is unweighted, any spanning tree is a minimum spanning tree. Therefore the efficiency of RAD depends on the size of a multicast group M only, which is decided by the topological position of the chosen *public routing proxy* p since M consists of all nodes within the HopCount (p, k) distance from p. We will show how to select good candidates for *public routing proxy* in more detail in the following section.

4. Selection of routing proxies

Algorithm 1. Finding public routing proxies

Require: $k \leqslant |V|$
1: **for all** $u \in V$ **do**
2: Initialize the set of routing proxies $P_u \leftarrow \emptyset$.
3: Initialize the set of recipient nodes $R_u \leftarrow \emptyset$.
4: Compute HopCount (u,k).
5: Compute CoverSize (u,k).
6: **end for**
7: **for all** $u \in V$
8: **for all** $v \in V$ **do**
9: Compute Distance (u,v).
10: **end for**
11: **end for**
12: **for all** $u \in V$ **do**
13: **for all** $v \in V$ **do**
14: **if** HopCount (u,k) > Distance (u,v) **then**
15: Add u into P_v.
16: Add v into R_u.
17: **end if**
18: **end for**
19: **end for**
20: $P = \cup_{v \in V} P_v$
21: Sort all *public routing proxies* $p \in P$ in descending order with the size of R_p.
22: **for all** $p \in P$ **do**
23: *removable*$[p] \leftarrow$ TRUE.
24: **for all** $r \in R_p$ **do**
25: **if** $|P_r| = 1$ **then**
26: *removable*$[p] \leftarrow$ FALSE.
27: **end if**
28: **end for**
29: **if** *removable*$[p]$ = TRUE **then**
30: **for all** $r \in R_p$ **do**
31: Delete p from P_r and set $R_p \leftarrow \emptyset$.
32: **end for**
33: **end if**
34: **end for**

In this section, we explain how to select a set of routing proxies of each possible recipient in detail. The selection algorithm needs to efficiently select *public routing proxies* for every vertex in V. Given a security parameter k, the *public routing proxies* for all network entities can be simultaneously selected by Algorithm 1. To explain this algorithm, we define "CoverSize" as follows:

Definition 4.1 (*CoverSize*). Let CoverSize (u,k) be the number of network nodes within the multi-hop communication scope of HopCount (u,k) where u is a public routing proxy and k is the parameter of k-anonymity.

Line 1–11 is the initialization step, computing HopCount (u,k) and CoverSize (u,k) for all possible recipients $u \in V$ and the given anonymity parameter k, and also computing Distance (u,v) that is the shortest path length of an arbitrary pair (u,v). For all $u \in V$, we initialize the set of routing proxies serving for u, P_u, and the set of recipient nodes with u as one of their *public routing proxies*, R_u. Note that each node can be *public routing proxy* and recipient node at the same time.

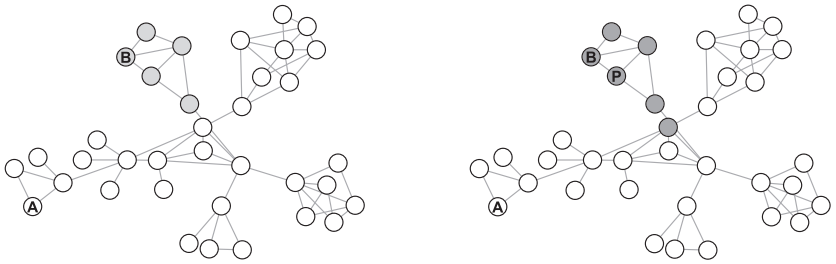
The procedure in line 12–19 will only add a vertex u into the *public routing proxies* P_v of each vertex v which are located within the distance HopCount (u,k) from u. In line 20, P is computed as the union of the sets P_v of *public routing proxies* for the senders $v \in V$. According to the definition of HopCount (u,k), we can see that for all $p \in P$ in line 20 serves for at least k nodes.

In the next step in line 21–34, we delete some redundant proxies from P. As we mentioned in Section 3, the efficiency of RAD depends on which *public routing proxy* is chosen. Therefore we need to delete improper proxy candidates which are likely to lead to a heavy traffic situation. The deletion of redundant proxies is also helpful to reduce the size of the *public routing proxies* to be maintained. We do this by repeatedly deleting "removable" proxy from P. A "removable" proxy $p^{\text{removable}}$ is denoted as redundant proxy where all the network entities with $p^{\text{removable}}$ as one of their *public routing proxies* have also at least another public proxy $\bar{p}(\neq p^{\text{removable}})$. This means that all the network entities associated to $p^{\text{removable}}$ still have a public proxy $\bar{p}$ even if $p^{\text{removable}}$ is deleted from their *public routing proxies*.

Here, we explain how to delete redundant proxies leading to a heavy traffic situation. The key observation is that a *public routing proxy* with a large CoverSize will lead to a heavy traffic situation in the protocol. In other words, a public routing proxy with the smallest CoverSize is preferred to reduce this message traffic as long as it guarantees recipient k-anonymity. Therefore, in line 23–28, we sequentially check whether $p \in P$ can be deleted in descending order with the CoverSize after sorting them. In line 29–34, for all $r \in R_p$, we delete p from P_r. For example, in Table 1, the node "6" is first tested to be deleted since the node "6" has the largest CoverSize (= 5). However the node "6" is not deleted since the node "6" is not "removable". Next, the node "1" is tested and then deleted since the node "1" is "removable".

Algorithm 1 guarantees that an intended recipient r correctly receives the message m in RAD if the graph is connected. First of all, we show that there exists at least one

Author's personal copy



(a) B's *public routing proxies* (light gray) (b) The actual recipients in multicast (dark gray)

Fig. 5. An example scenario of 5-anonymity data delivery by RAD: the sender **A** sends a message to the recipient **B** via one of **B**'s *public routing proxies*, **P**.

Table 1
An example of deleting redundant *public routing proxies*.

i	Before deleting the node "1"		After deleting the node "1"	
	P_i	R_i	P_i	R_i
1	{1,2,5,6}	{2,3,5}	{2,5,6}	
2	{1,2,6}	{1,2,3}	{2,6}	{1,2,3}
3	{1,2,4,6}	{}	{2,4,6}	{}
4	{4,5,6}	{3,4,6}	{4,5,6}	{3,4,6}
5	{1,4,5}	{1,4,5}	{4,5}	{1,4,5}
6	{6}	{1,2,3,4,6}	{6}	{1,2,3,4,6}

public routing proxy for r. In line 12–19, at least all neighbours of r should be added into P_r if $k > 1$. Since there exists $p \in P_r$ such that *removable*[p] = FALSE, r always has at least one *public routing proxy*. Thus RAD is always successfully terminated since the distance between the intended recipient r and its a *public routing proxy* $p \in P_r$, which is chosen by this algorithm, is less than HopCount (p,k).

The time complexity of Algorithm 1 is $O(n^3)$ time. Using breadth-first search, we can compute HopCount (u,k) and CoverSize (u,k) for $u \in V$ in $O(n)$ time. Therefore, the computation in line 1–6 can be processed in $O(n^2)$. In line 7–11, we compute the shortest distance between every pair of vertices in $O(n^3)$. For the addition of *public routing proxies*, the computation in line 12–19 can be processed in $O(n^2)$ time. In line 20–21, we sort P in $O(n\log n)$ time. Finally, for the removal of *public routing proxies*, the computation in line 22–34 can be processed in $O(n^2)$ time. Consequently, the total running time is in $O(n^3)$.

5. Analysis

In this section, we empirically analyse the performance and anonymity of RAD with several network datasets.

5.1. Performance

We define two following cost functions, "$\text{Cost}_{\text{Latency}}$" and "$\text{Cost}_{\text{Traffic}}$" in Definitions 5.1 and 5.2, to measure the performance.

Definition 5.1 ($Cost_{Latency}$). The number of rounds to deliver a message to the intended recipient required by the protocol, presuming that every network entity is allowed to send arbitrarily many messages with every (global) time tick and to receive arbitrarily many messages sent by other entities at the beginning of a round.

Definition 5.2 ($Cost_{Traffic}$). The total number of messages transmitted among all network entities according to the protocol.

We approximate the expected values of these costs for RAD. Given the graph G and the security parameter k, let $L(G)$,$H(G,k)$ and $D(G)$ be the average shortest path length among all pairs of vertices, the average HopCount and the average degree, respectively. The upper bound of $\text{E}[\text{Cost}_{\text{Latency}}]$ is simply derived in Eq. (1). Here, $\text{E}[X]$ denotes the expected value of the argument random variable X.

$$\text{E}[\text{Cost}_{\text{Latency}}] \leqslant L(G) + H(G,k) \tag{1}$$

This is because the average hop count from a source s to the public routing proxy p of an intended recipient r is $L(G)$ and the average hop count from p to r is $H(G,k)$.

The upper bound of $\text{E}[\text{Cost}_{\text{Traffic}}]$ is also derived in Eq. (2). Here, the number of leaves at depth h is d^h in the complete d-ary tree.

$$\begin{aligned}\text{E}[\text{Cost}_{\text{Traffic}}] &\leqslant L(G) + \sum_{i=1}^{H(G,k)} D(G)^i \\ &= L(G) - 1 + \frac{D(G)^{(H(G,k)+1)} - 1}{D(G) - 1}\end{aligned} \tag{2}$$

In Eq. (2), we expect the upper bound of $\text{Cost}_{\text{Traffic}}$ to be linearly proportional to the security parameter k since $H(G,k) \leqslant \log_{D(G)} k + 1$ in a complete $D(G)$-ary tree.

We evaluated the performance of our RAD protocol on several network datasets in order to show that our scheme can perform well in many network topologies as follows: four random graphs (Random_{001}, Random_{005}, Random_{01}, Random_{02} – we denote as Random_p the random graph with the linking probability p, which means the probability that two vertices are directly connected via an edge.), two Transit-Stub graphs [4], a ring graph, a Watts-Strogatz small world graph [37], a Barabási-Albert scale free network [2], a Content-Addressable Network (CAN) [26], a Chord network [33], and a Hypergrid graph [29]. The network topologies are shown in Fig. 6. Inherently, the hop count for multicast to satisfy k-anonymity property will be varied depending on the underlying network topologies. For example, in a network of linear chain, a hop count should be at least k. However, at a *public routing proxy* with k neighbors, one hop count is enough for k-anonymity. To show the effect of network topologies, we analysed the performances of our RAD protocol on various network

Author's personal copy

topologies. We summarize the properties of the networks used in the experiments in Table 2. Network density is a normalized version of the average number of neighbours, which indicates the average connectivity of a node in a network. Network diameter is the maximum distance between nodes in the network [21]. Network density describes the overall level of interaction between all nodes in a network [11].

Our experimental scenarios come under two broad categories. In the first category ($Random_{001}$,$Random_{005}$, $Random_{01}$,$Random_{02}$), we analysed the cost/efficiency of RAD depending on the effect of network density by varying the linking probability from 0.001 to 0.2. In the second category (the remaining networks), we also analysed the cost/efficiency of RAD on various network topologies.

We randomly generated 300 routing queries between a sender and a recipient for each network. We tested the cost/efficiency by varying k from 10 to 100. Note that the preprocessing cost for setting *public routing proxies* is ignored. The cost of this preprocessing requires $O(n^3)$ computation time and $O(n)$ messages to distribute *public routing proxies* where n is the number of network entities. For comparison purposes, we implement three multicast protocols that generate a multicast tree for the recipient set consisting of the randomly selected $k-1$ network entities and the intended recipient r: (i) Nearest Participant First (NPF) algorithm [35], (ii) the Kou, Markowsky and Berman (KMB) algorithm [17], and (iii) finding the Shortest Path Tree between the sender and each of the recipient set (SPT). These three heuristics were designed to focus on the construction of efficient multicast trees only. Therefore, given a set of recipients, these heuristics have been shown to perform well in practice [24,38]. Unlike these heuristics, our scheme is designed to focus not only on the construction of efficient multicast trees but also the selection of optimized dummy nodes, as discussed in Section 1. Note that since our traffic and security assumption is totally different from the Chaum's *mix* scheme or *Onion* routing (as discussed in Section 2.1), we do not compare our scheme with Chaum's.

Fig. 7 shows the $Cost_{Traffic}$ of each protocol. Our proposed protocol RAD always produced better results for all networks except $Random_{02}$ (when $k < 60$) than the other three schemes (i.e., NPF, KMB, and SPT). In particular, RAD performs well for the networks with a low network density. We can see that the $Cost_{Traffic}$ of RAD is significantly lower than the other protocols for Ring, Chord and Hypergrid (see Fig. 7(g), (k) and (l)).

Fig. 8 shows the $Cost_{Latency}$ of each protocol. The SPT heuristic serves as an absolute lower bound since it provides the shortest path between the sender and the intended recipient. RAD is the second-best protocol for most networks although RAD produced the worst results for two Transit-Stub graphs. Moreover, we can see that RAD scales well with the security parameter k except Transit-Stub graphs. We found the pattern of fluctuation for Ring and Chord (see Fig. 8(g) and (k)). Interestingly, these results' patterns are similar to those of the SPT heuristic.

Our experimental results show that RAD has the trade-off between the $Cost_{Traffic}$ and the $Cost_{Latency}$ compared with SPT. In order to show this more effectively, we computed the ($Cost_{Traffic}$ of RAD)/($Cost_{Traffic}$ of SPT) and ($Cost_{Latency}$ of RAD)/($Cost_{Latency}$ of SPT), respectively, when k=10, 50 and 100. The results are shown in Table 3. As shown in Table 3, for k = 10 and Chord network, RAD uses only 27% traffic of SPT without a significant increase in $Cost_{Latency}$ compared to the shortest path length. When we use P2P (e.g., Chord) as transport network, this result will be very encouraging.

Finally, we can see the size of the *public routing proxies*, which is computed by Algorithm 1, for each network entity. The experimental results are shown in Fig. 9. We can see that the average size of the *public routing proxies* for each network entity is scalable for a large k. Therefore, $O(n)$ space is enough to maintain the *public routing proxies* for all network entities since $O(1)$ space is approximately required to store the *public routing proxies* for a network entity. However, we can also see that the sizes of the *public routing proxies* for some network entities is relatively large compared to those of the other entities. In particular, this situation is clearly shown in two Transit-Stub graphs and a scale free network. For example, in a Transit-Stub graph, the maximum size of the *public routing proxies* for a network entity is greater than roughly 6.7 times than the average size of that when k = 30 (see Fig. 9(e)). In the worst case, however, the size of the *public routing proxies* for a network entity can be increased to n although the mean value for all network entities is bounded by a constant. We will discuss this limitation in more detail in Section 6.

5.2. Anonymity

We should consider two attack scenarios: "message tracing attacks" and "attacks with *public routing proxies*".

Basically, in RAD, any message tracing attack is not effective. Let R be the set of vertices (including the intended recipient r) within HopCount (p,k) distance in RAD. We note that every network entity in R has the same possibility to be the recipient r since the multicast protocol is processed under the same rule. By the definition of HopCount (p,k),$|R|$ is at least k. In other words, no one can infer the information about the intended recipient r with a probability greater than $1/k$ except the sender and the intended recipient themselves.

For "attacks with *public routing proxies*", attackers can use the information about *public routing proxies*. Let A be the set of the network entities with p as one of their *public routing proxies*. Although it seems helpful to reduce the number of candidates for r since $|A| \leqslant |R|$, we still guarantee that $|A|$ is at least k. In order to show this, we prove that p should serve for at least k nodes as follows. In Algorithm 1, p is added into the *public routing proxies* of at least k nodes, respectively, by the definition of HopCount (p,k) and never deleted. If p is "removable", p should be deleted in the course of Algorithm 1. Therefore, $|A| \geqslant k$. We show that an unlimited attacker cannot distinguish the recipient r from a network entity x in A with a probability greater than 1/2. First of all, both of r and x have equally p, which was used in the message delivery, as one of their *public routing proxies*. In other words, the attacker cannot gain any information about r from p. In addition, since the proxy

Author's personal copy

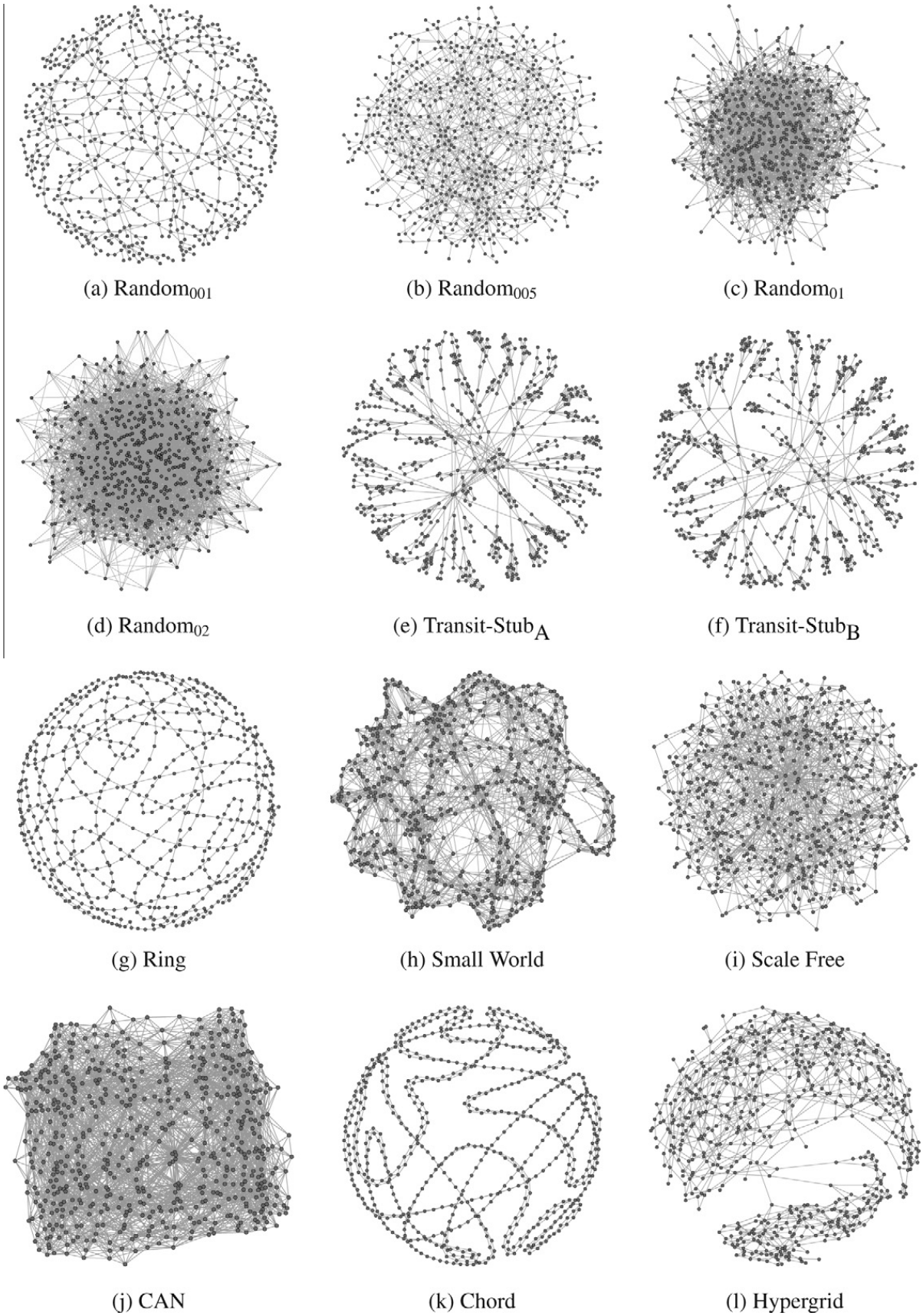


Fig. 6. The networks used in the experiments.

p distributes m to every entity in A in the exactly same manner, the attacker cannot also gain any information about r. Therefore any network entity in A has the same possibility to be the recipient r.

Author's personal copy

Table 2
Some properties of the networks used in our experiments.

Network	♯ of vertices	♯ of edges	Diameter	Density	$D(G)$	$L(G)$
$Random_{001}$	594	623	35	0.004	2.098	13.334
$Random_{005}$	584	929	13	0.005	3.182	5.786
$Random_{01}$	599	1,778	7	0.010	5.937	3.796
$Random_{02}$	600	3,547	5	0.020	11.823	2.845
$Transit\text{-}Stub_A$	600	1,164	14	0.006	3.880	7.078
$Transit\text{-}Stub_B$	600	1,228	14	0.007	4.093	7.060
Ring	600	600	300	0.003	2.000	150.250
Small World	600	3,000	7	0.017	10.000	4.208
Scale Free	600	1,182	7	0.007	3.940	3.986
CAN	600	5,401	9	0.030	18.003	4.283
Chord	600	1,200	150	0.007	4.000	75.376
Hypergrid	600	1,099	11	0.006	3.663	7.318

When attackers combine two attack scenarios, "message tracing attacks" and "attacks with *public routing proxies*", there is still no advantage compared with "attacks with *public routing proxies*" since $A \cap R = A$.

By using the concept of entropy [32] we can measure the anonymity of RAD in a quantitative manner. Let $P(A)$ be the attacker's the posteriori probability distribution of A. For our RAD protocol, the entropy $H_{P(A)}$ can be simply defined depending on $|A|$ only as $H_{P(A)} = \log(|A|)$ (here, log is to the base 2) [23] since every network entity in A is a plausible recipient with the same probability. Thus, *anonymity set* must have at least log$k$ entropy in our RAD protocol. As evidence, Fig. 10 shows the degree of anonymity for our RAD protocol.

We calculate the entropy values of the minimum/maximum/average anonymity sets for each network dataset in Section 5.1. For comparison, we plotted the values against log$k$ as an absolute lower bound. As shown in Fig. 10, as the size of k increases, the degree of entropy steadily increases in all kinds of networks. This is because the size of *anonymity set* is proportional to k. We can see that RAD always provides a higher entropy than the theoretical lower bound, log k, required for k-anonymity.

In general *correlation attacks* are also not effective. There is no specific correlation information between the sender and the intended recipient since the sender only delivers the message m to *public routing proxy* p and then p distributes the message to the intended recipient and the dummies in the exactly same manner.

6. Limitations

In Section 5.1, we observed that the size of *public routing proxies* for each network entity is not bounded by a specific constant. Since this size can be theoretically increased to n in the worst case, $O(n^2)$ space is required for a network entity to store all *public routing proxies*. Unfortunately, it is not easy to bound the maximum size of the *public routing proxies* for a network entity within a constant. When we consider this, we note that finding *public routing proxies* to optimally minimize routing cost is NP-hard. In general this problem can be reduced to a variant of ρ-median problem[1] which is known to be NP-hard [15]. The ρ-median problem on a graph is to identify a subset $S \subseteq V$, called medians, where $|S| = \rho$ so as to minimize $\sum_{v \in V} \min_{s \in S} d(v, s)$. Given a graph $G = (V, E)$, a security parameter k and a constant c, our goal is to identify *public routing proxies* with the size c for each network entity. Unlike the standard ρ-median problem, this requires that $|V_p| \geqslant k$ for any *public routing proxy* p where V_p is the set of entities with p as their *public routing proxy*.

We also consider only the static network model. In dynamic networks, it is practically more difficult to maintain each network entity's *public routing proxies* to guarantee k-anonymity due to the entity's mobility. However, we can capture well the statistical property (e.g. the average node degree over all time) of a dynamic network and use this information to compute reasonable *public routing proxies*. In this case, the computed *public routing proxies* may provide k-anonymity with a high probability. We will consider extending our work to dynamic networks as important lines for future work.

Our RAD is designed only for one-way k-recipient-anonymous data delivery. For two-way communication, we need to consider timing attacks. This is because an attacker may infer that two entities are communicating by observing the incoming and outgoing traffic at their *public routing proxies* and what the nodes reached via these proxies. We can design additional mechanisms such as latency schedule to support the two-way communication against timing attacks. We leave this support of two-way k-recipient-anonymous communications as future work.

Finally, our RAD is practically hard to deploy only with legacy multicasting in large-scale networks, such as the Internet. In the Internet, many networks are independently managed as Autonomous Systems (ASes) by different administrators, so their internal network structures are rarely known to other ASes. To support the recipient k-anonymity in this large-scale Internet, we can use P2P approaches running on overlay networks implemented at the application layer. Those nodes interested in this recipient k-anonymity subscribe to the P2P network for the k-recipient-anonymous data delivery. With this P2P approach, we can deploy our RAD protocol in the large-scale networks, such as the Internet, without the centralized administration. Also, note that the number of the active nodes in the P2P network should be at least k in the anonymity set of a proxy for the recipient k-anonymity. Since churns in the P2P network happen due to nodes' join to

[1] This problem is commonly referred to as the p- or k-median problem. To avoid the confusion with p and k which are used for other purposes in this paper, we use the symbol ρ.

Author's personal copy

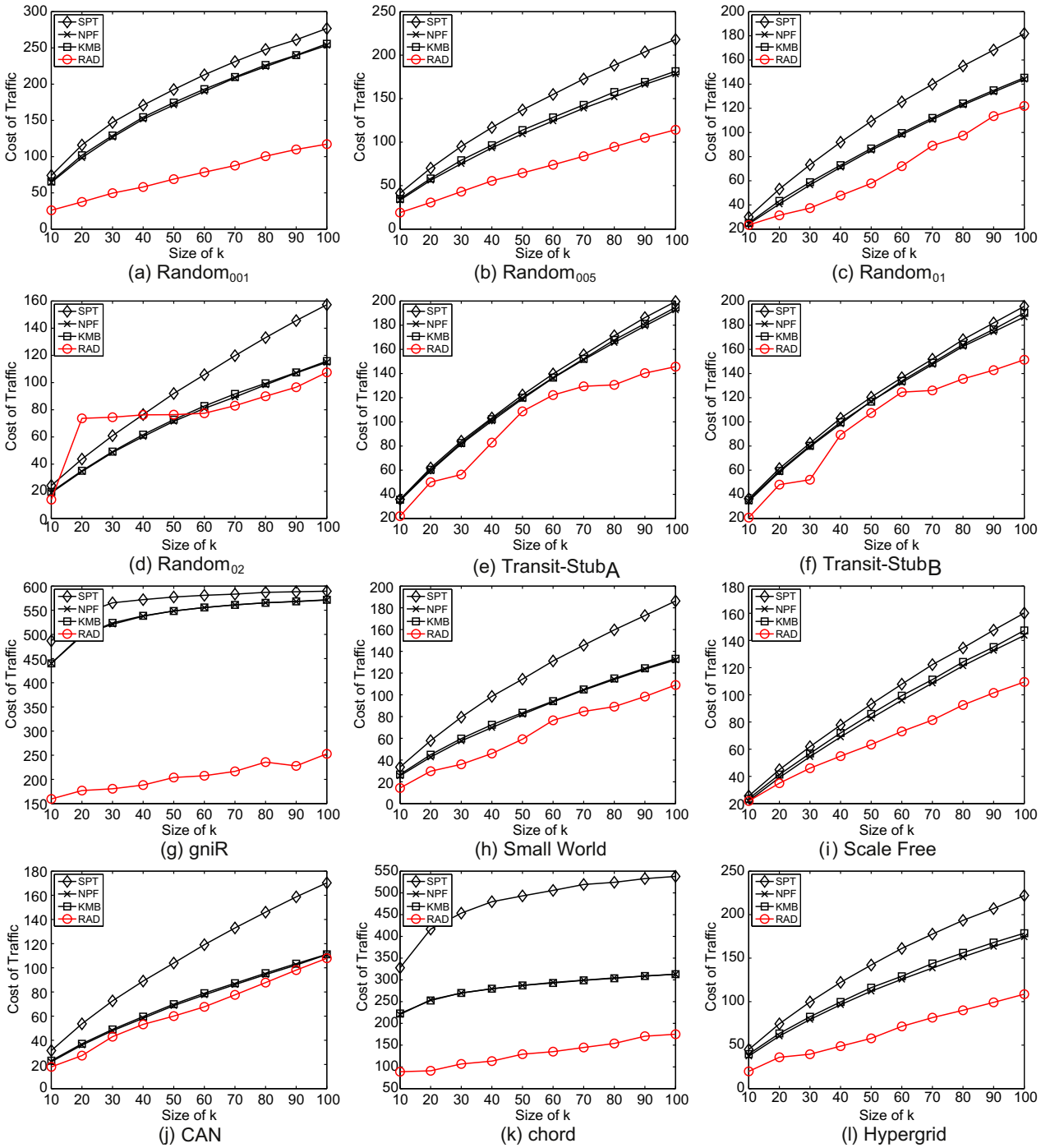


Fig. 7. Cost of traffic.

and departure from the network, it is necessary to select enough nodes even under this churn activity such that the average number of active nodes is close to the security parameter k. This extension of our RAD to the P2P networking (including the churn) is left as future work.

7. Related work

Pfitzmann and Köhntopp [23] introduced a set of informal definitions that characterizes anonymity threats in networks. Information theoretic anonymity metrics [9,31] based on the concept of entropy provide to quantify the degree of anonymity in traffic analysis.

There has been a lot of interest in anonymous communication. Chaum [5] introduced the concept of a network of *mixes*. A *mix* is an intermediate network entity that takes a batch of messages, performs cryptographic operations on the messages, and then forwards the shuffled messages to the next destination. An ideal *mix* is to adequately hide the relationship between incoming messages and outgoing messages at the *mix*. In practice, however, the use of multiple

Author's personal copy

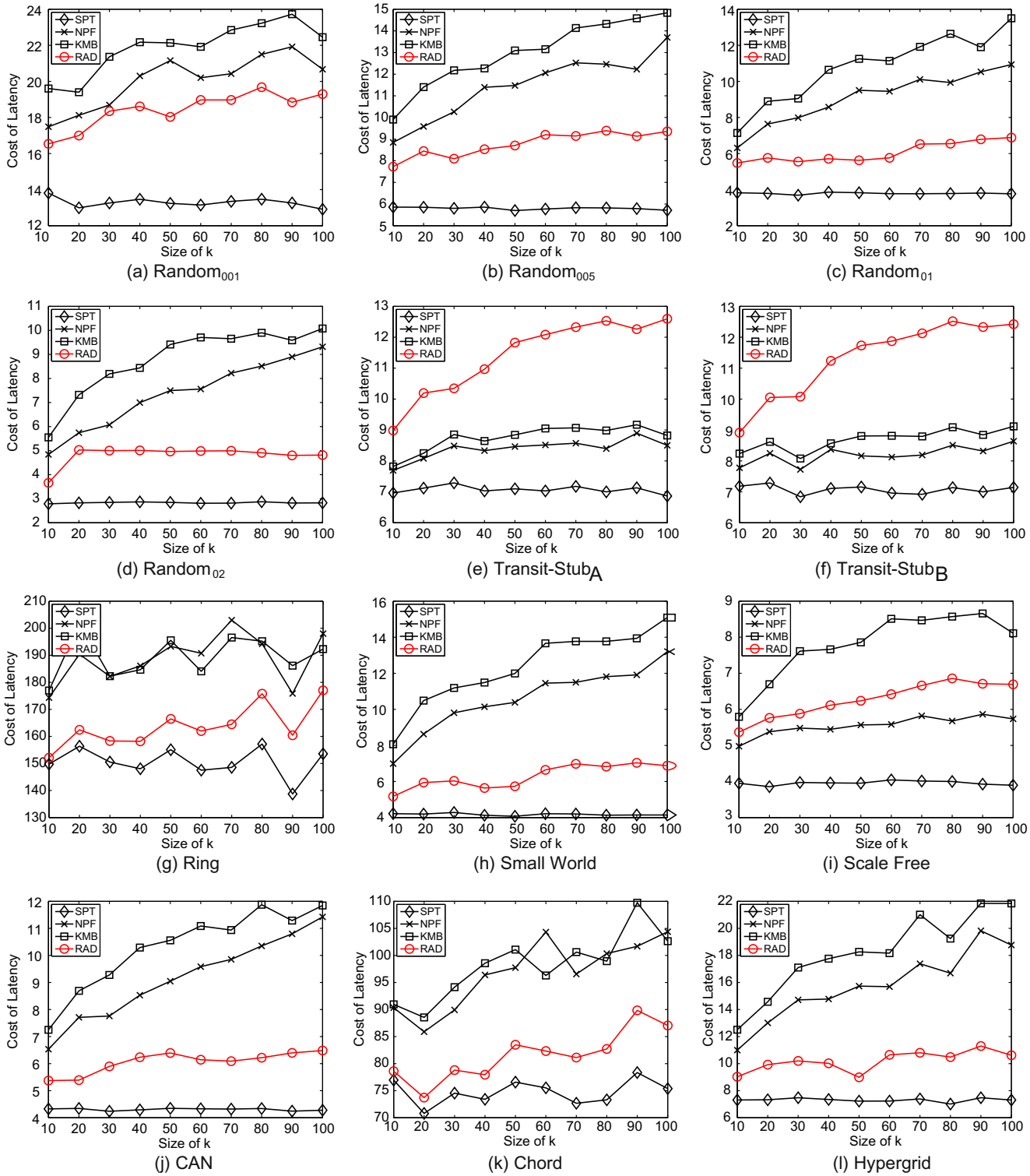


Fig. 8. Cost of latency.

mixes is more desirable since a single *mix* may be compromised. In this approach, sufficient network traffic is necessarily needed at a *mix* to scramble them in order to resist traffic analysis. When the traffic is not enough at the *mix*, the message forwarding may be delayed until sufficient traffic is collected or fake messages may be required. Rackoff and Simon [25] proposed a system based on *mixes*. There are several variants of the original system [3,8,14,19,20].

Reed et al. [27] proposed *Onion routing* that is a framework based on multiple *mixes* in a way that allows various applications for anonymous communication (or connection). In Onion routing, a path consisting of randomly chosen onion routers is constructed for anonymous

Author's personal copy

Table 3
Ratio of RAD and SPT. The $Cost_{traffic}$ ratios are highlighted in bold font.

Network	$k = 10$		$k = 50$		$k = 100$	
	$Cost_{Traffic}$	$Cost_{Latency}$	$Cost_{Traffic}$	$Cost_{Latency}$	$Cost_{Traffic}$	$Cost_{Latency}$
$Random_{001}$	**0.3518**	1.1977	**0.3582**	1.3621	**0.4236**	1.4947
$Random_{005}$	**0.4604**	1.3187	**0.4709**	1.5269	**0.5237**	1.6381
$Random_{01}$	**0.7790**	1.4316	**0.5294**	1.4695	**0.6703**	1.8274
$Random_{02}$	**0.5736**	1.3134	**0.8308**	1.7436	**0.6837**	1.7028
$Transit\text{-}Stub_A$	**0.6118**	1.2901	**0.8889**	1.6665	**0.7299**	1.8354
$Transit\text{-}Stub_B$	**0.5700**	1.2408	**0.8913**	1.6395	**0.7740**	1.7392
Ring	**0.3278**	1.0158	**0.3531**	1.0735	**0.4292**	1.1529
Small World	**0.4266**	1.2269	**0.5175**	1.4075	**0.5853**	1.6589
Scale Free	**0.8548**	1.3586	**0.6806**	1.5776	**0.6835**	1.7177
CAN	**0.5726**	1.2419	**0.5776**	1.4709	**0.6335**	1.5160
Chord	**0.2711**	1.0214	**0.2620**	1.0902	**0.3257**	1.1548
Hypergrid	**0.4417**	1.2353	**0.4066**	1.2434	**0.4890**	1.4530

communication from a sender to a recipient and then the sender creates a layered message which is recursively encrypted with the onion routers' public keys. Each onion router on the path removes a layer of encryption from the encrypted message with its private key and then delivers the message to its successor (an onion router or the intended recipient) on the path. Tor network [10] is a widely used, general purpose platform for Onion routing. Xiao et al. [39] proposed a scheme for anonymous multicast communication by extending *Onion routing*. In general, however, it cannot guarantee k-anonymity that is our security goal: consider a single user using an Onion routing system – there are no k users to hide the intended recipient among, so it fails to achieve k-anonymity. Also, it is not effective against *correlation attacks* (e.g. using the end-to-end timing information between the sender and the recipient) [18]. Finally, unlike ours, there is the performance overhead associated with cryptographic operations based on public key cryptography. In order to construct an anonymity routing path for Tor, a user should share a unique key with each onion router on the path. It requires $2 \cdot i$ messages to share the key with the ith onion router on the path. Thus the total number of messages required to construct an anonymity routing path of length l is $l \cdot (l+1) (= 2 \cdot \sum_{i=1}^{l} i)$. Note that each of these messages needs to be encrypted and decrypted at each hop. Thus it is difficult to be implemented and deployed n networks requiring high-performance data forwarding, such as the Internet.

Another important approach to anonymous communication is Crowds [28]. Crowds is a set of network entities that forward an entity's message through a path between crowd members. Message traffic is encrypted and forwarded to a crowd member instead of the intended recipient. This crowd member forwards it either to another crowd member or to the intended recipient directly. However, this makes communication resistant to local observer only. This approach does not protect against an outside adversary who monitors all messages traffic.

Our study is partly motivated by such problems with Onion routing and Crowds. We propose the protocol to achieve unconditional k-anonymity without any cryptographic primitives for high performance. Farber and Larson [13] presented that broadcast can be used for unconditional anonymous communication. Over the last decades, however, it is commonly believed that this approach is impractical since it requires extremely expensive traffic overhead. Adelsbach and Greveler [1] revisited the broadcast-based anonymous communication in the case of wireless networks where broadcast communication is basically supported on the data link layer. They discussed the practicality of the broadcast-based anonymous communication by implementing a simple prototype.

In this context, multicasting can be inherently applied to provide *recipient anonymity*. Algorithms for constructing multicast trees have been developed with two optimization goals: minimization of tree height and/or tree size. A minimum height tree can be constructed in $O(n^2)$ time using shortest path algorithm, where n is the number of nodes in the network. The problem to find a multicast tree with minimum size is called the *Steiner tree problem* which is known to be NP-hard [16], even if edges have unit cost. Several heuristic algorithms were proposed to find approximate Steiner trees [17,35]. These algorithms run in $O(n^2)$ and $O(n^3)$. Empirical analyses show that these heuristics produce near-optimal trees in most cases. Moreover, they guarantee a constant-factor approximation. We show that RAD perform better than these heuristics in reducing the message traffic with various network topologies. For example, RAD uses the traffics ranging from 27% to 85% of the shortest path-based multicasting with a little sacrifice of delay, that is, the increased latency varied from 1.0 to 1.43 times compared to the multicasting.

8. Conclusion

In this paper, we proposed a new anonymous communication protocol for *recipient anonymity* by introducing *public routing proxy*. Our key insight is to deliver a message to a routing proxy topologically relevant to the intended recipient so that the proxy can multicast the message to a set of proximate network entities including the original recipient. The legacy scheme uses the broadcasting for this *recipient anonymity* where messages are broadcasted to all possible network entities, leading to the expensive traffic overhead. In our proposed protocol RAD, the user-required security level k can be flexibly set, depending on the user's

Author's personal copy

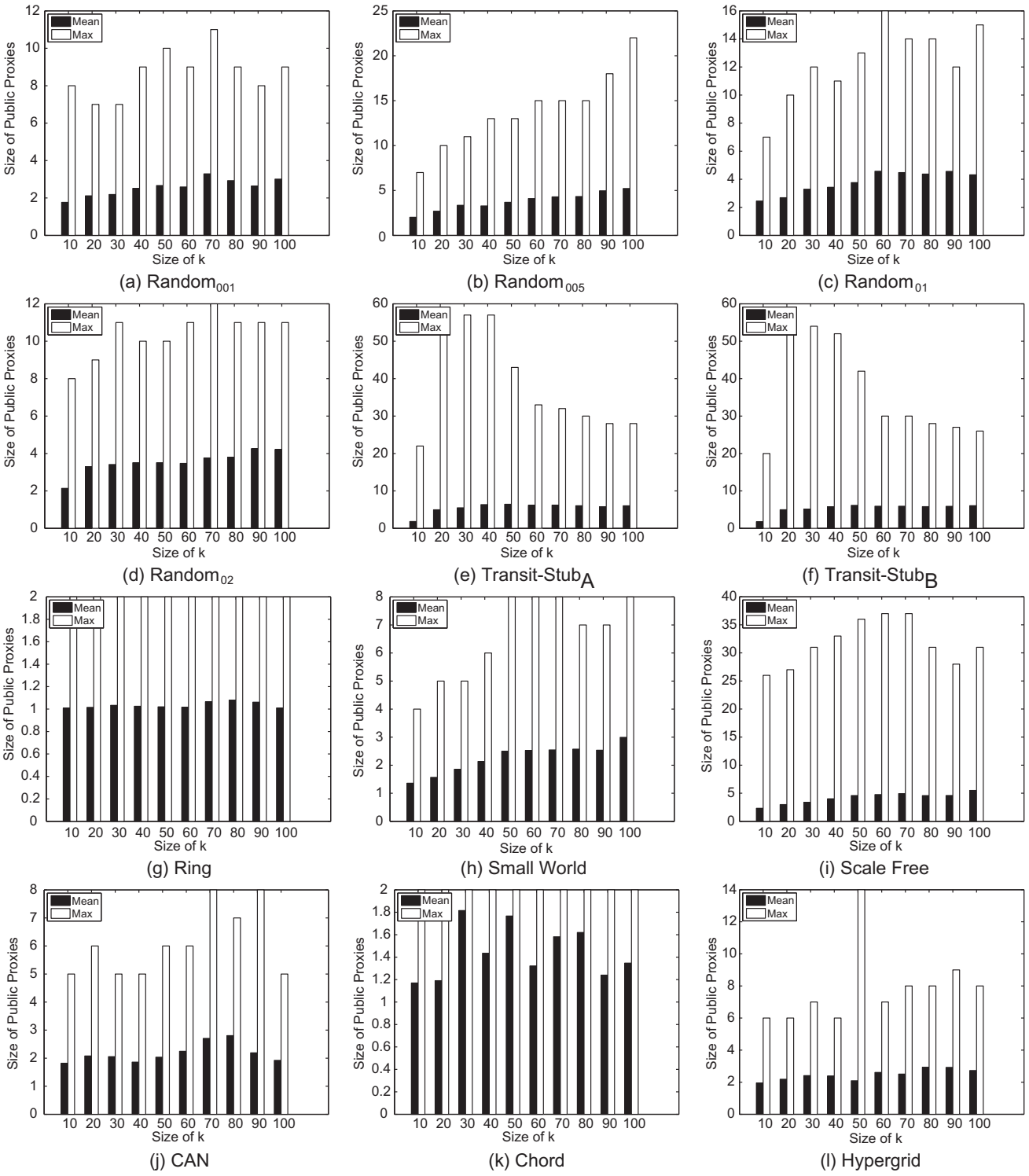


Fig. 9. Size of public routing proxies.

requirements for anonymity and efficiency. We showed that the proposed protocol RAD can provide unconditional security for recipient k-anonymity and at the same time the reasonable efficiency can be achieved in terms of message traffic and latency. We demonstrated the performance of the protocol from the theoretical and empirical analysis. We evaluated the performance of RAD by comparing it to three different heuristics on several network datasets in order to show that our scheme can perform well in many network topologies (including the Internet) and demonstrated that RAD is capable of achieving a low traffic overhead and a reasonable network latency.

We specify civil applications and military applications based on one-way communications for our recipient-anonymity based data delivery. For the civil applications, P2P networks can adopt our RAD protocol to send some

Author's personal copy

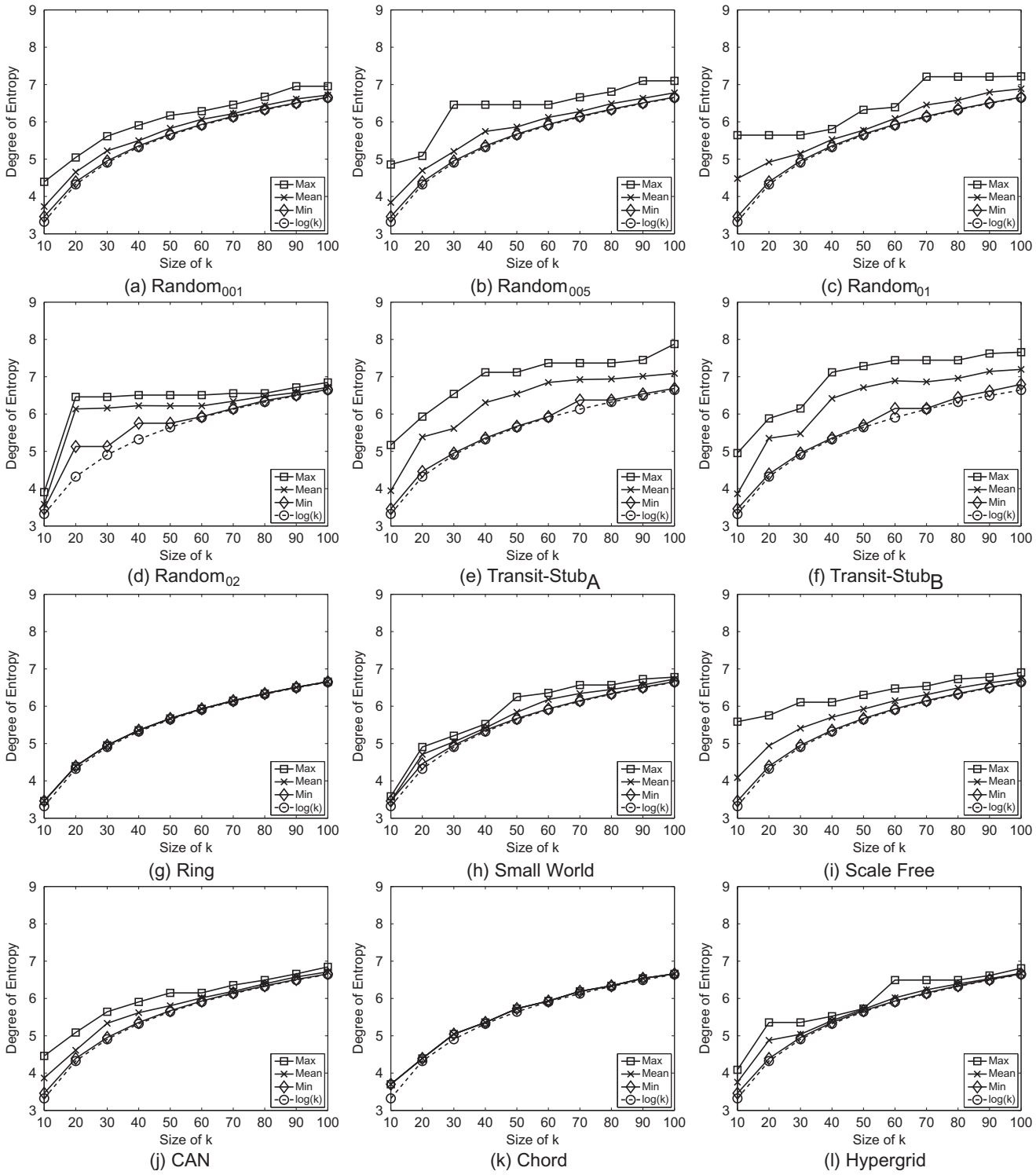


Fig. 10. Degree of anonymity for our protocol.

patients their privacy-related advertisement messages, such as medicine or food advertisement for health. For the military applications, the commander center can deliver the military operation commands to soldiers or policemen under military operations such that the adversaries cannot tell which people are soldiers or policemen near them. However, the current RAD does not come without limitations for these applications. It involves the preprocessing phase, where the administrator computes the *public routing proxies* of network entities. This task would have to be carried out periodically as the network topology might change from time to time, affecting the performance and the anonymity of RAD protocol. As part of the future work, we plan to extend the RAD protocol for highly mobile networks such as mobile ad hoc networks and evaluate its performance and anonymity.

Author's personal copy

For a high traffic density network, our RAD protocol produces high traffic overhead compared to the *mix* schemes and *Onion* routing. This is because our RAD protocol is designed regardless of the overall network traffic flow unlike the conventional anonymous protocols. In fact, the performance of these protocols highly depends on the volume of network traffic continuously generated; perhaps, this idea can also be applied to our RAD protocol. Therefore, we need to consider the network traffic volume for a better anonymity protocol. A possible approach is to use a hybrid strategy that switches between multicast and *mix*, depending on the amount of traffic. This is because RAD decreases the delivery latency compared with the *mix* schemes and *Onion* routing while it produces relatively high traffic volume. Future work may explore the possibility of a hybrid approach in depth to reduce the traffic overhead.

Acknowledgements

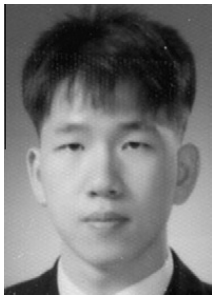
The authors thank Jon Crowcroft and the anonymous referees for their careful attention and insightful comments.

References

[1] André Adelsbach, and Ulrich Greveler, ANOCAST: Rethinking Broadcast Anonymity in the Case of Wireless Communication. In Sicherheit, pages 71–84, 2008.
[2] Albert-László Barabási, Réka Albert, Emergence of Scaling in Random Networks, Science 286 (5439) (1999) 509–512.
[3] Oliver Berthold, Hannes Federrath, and Stefan Köpsell, Web MIXes: a system for anonymous and unobservable Internet access, In International workshop on Designing privacy enhancing technologies, Springer-Verlag New York, Inc, New York, NY, USA, 2001, 115–129
[4] K.I. Calvert, M.B. Doar, E.W. Zegura, Modeling Internet Topology, Communications Magazine, IEEE 35 (6) (1997) 160–163.
[5] David L. Chaum, Untraceable electronic mail, return addresses, and digital pseudonyms, Communications of the ACM 24 (2) (1981) 84–90.
[6] Jaihyung Cho, James Breen, Analysis of the performance of dynamic multicast routing algorithms, Computer Communications 22 (7) (1999) 667–674.
[7] George Danezis. Introducing Traffic Analysis: Attacks, Defences and Public Policy Issues.. (Invited Talk). December 2005.
[8] Yvo Desmedt, and Kaoru Kurosawa, How to Break a Practical MIX and Design a New One, In EUROCRYPT, pages 557–572, 2000.
[9] Claudia Díaz, Stefaan Seys, Joris Claessens, Bart Preneel, Towards Measuring Anonymity, In Privacy Enhancing Technologies (2002) 54–68.
[10] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-Generation Onion Router. In Proceedings of the 13th USENIX Security Symposium, pages 303–320, 2004.
[11] J. Dong, S. Horvath, Understanding Network Concepts in Modules, BMC Systems Biology 1 (1) (2007).
[12] Carl Ellison, Bruce Schneier, Ten Risks of PKI: What You're not Being Told about Public Key Infrastructure, Computer Security Journal 16 (1) (2000) 1–7.
[13] David J. Farber and Kenneth C. Larson, Network Security Via Dynamic Process Renaming. In Fourth Data Communications Symposium, pages 8–18, Quebec City, Canada, October 1975.
[14] Markus Jakobsson. Flash mixing. In PODC '99: Proceedings of the eighteenth annual ACM symposium on Principles of distributed computing, pages 83–89, New York, NY, USA, 1999. ACM.
[15] O. Kariv, S.L. Hakimi, An Algorithmic Approach to Network Location Problems. I: The p-Centers, SIAM Journal on Applied Mathematics 37 (3) (1979) 513–538.
[16] R.M. Karp, Reducibility Among Combinatorial Problems, In Complexity of Computer Computations, Plenum Press, 1972.
[17] L. Kou, G. Markowsky, L. Berman, A fast algorithm for steiner trees, Acta Informatica 15 (2) (1981) 141–145.
[18] Steven J. Murdoch, George Danezis, Low-Cost Traffic Analysis of Tor, In Proceedings of the 2005 IEEE Symposium on Security and Privacy, IEEE Computer Society, Washington, DC, USA, 2005.
[19] Miyako Ohkubo, Masayuki Abe, A Length-Invariant Hybrid Mix, In ASIACRYPT '00: Proceedings of the 6th International Conference on the Theory and Application of Cryptology and Information Security, Springer-Verlag, London, UK, 2000.
[20] Choonsik Park, Kazutomo Itoh, and Kaoru Kurosawa, Efficient anonymous channel and all/nothing election scheme, In EUROCRYPT '93: Workshop on the theory and application of cryptographic techniques on Advances in cryptology, pages 248–259, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.
[21] H. Per, H. Frank, Eccentricity and Centrality in Networks, Social Networks 17 (1) (1995) 57–63.
[22] Charles Perkins and Elizabeth Royer. Ad-hoc On-Demand Distance Vector Routing. In Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, pages 90–100, 1997.
[23] Andreas Pfitzmann and Marit Köhntopp. Anonymity, Unobservability, and Pseudonymity - A Proposal for Terminology. In Workshop on Design Issues in Anonymity and Unobservability, pages 1–9, 2000.
[24] Maciej Piechowiak, Piotr Zwierzykowski, and Sławomir Hanczewski. Performance Analysis of Multicast Heuristic Algorithms. In Third International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks, pages 41/1–41/8, 2005.
[25] Charles Rackoff, Daniel R. Simon, Cryptographic defense against traffic analysis, In STOC '93: Proceedings of the twenty-fifth annual ACM symposium on Theory of computing, ACM, New York, NY, USA, 1993.
[26] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, Scott Schenker, A Scalable Content-Addressable Network, In SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, ACM, New York, NY, USA, 2001.
[27] Michael G. Reed, Paul F. Syverson, David M. Goldschlag, Anonymous Connections and Onion Routing, IEEE Journal on Selected Areas in Communications 16 (1998) 482–494.
[28] Michael K. Reiter, Aviel D. Rubin, Crowds: anonymity for Web transactions, ACM Transactions on Information and System Security 1 (1) (1998) 66–92.
[29] Fabrice Saffre, Robert Ghanea-Hercock, Beyond anarchy: self-organized topology for peer to peer networks, Complexity 9 (2) (2003) 49–53.
[30] L.H. Sahasrabuddhe, B. Mukherjee, Multicast routing algorithms and protocols: a tutorial, Network, IEEE 14 (1) (2000) 90–102.
[31] Andrei Serjantov, George Danezis, Towards an information theoretic metric for anonymity, In PET'02: Proceedings of the 2nd international conference on Privacy enhancing technologies, Springer-Verlag, Berlin, Heidelberg, 2003.
[32] C.E. Shannon, A mathematical theory of communication, ACM SIGMOBILE Mobile Computing and Communications Review 5 (1) (2001) 3–55.
[33] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, Hari Balakrishnan, Chord: a scalable peer-to-peer lookup protocol for internet applications, IEEE/ACM Transactions on Networking 11 (1) (2003) 17–32.
[34] Latanya Sweeney, k-anonymity: a model for protecting privacy, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10 (5) (2002) 557–570.
[35] Hiromitsu Takahashi, Akira Matsuyama, An Approximate Solution for the Steiner Problem in Graphs, Mathematica Japonica 24 (1980) 573–577.
[36] Luis von Ahn, Andrew Bortz, Nicholas J. Hopper, k-anonymous message transmission, In CCS '03: Proceedings of the 10th ACM conference on Computer and communications security, ACM, New York, NY, USA, 2003.
[37] Duncan J. Watts, Small Worlds: The Dynamics of Networks between Order and Randomness (Princeton Studies in Complexity), illustrated ed., Princeton University Press, 2003.
[38] B.M. Waxman, Routing of multipoint connections, Selected Areas in Communications, IEEE Journal on 6 (9) (1988) 1617–1622.

Author's personal copy

[39] Li Xiao, Yunhao Liu, Wenjun Gu, Dong Xuan, Xiaomei Liu, Mutual anonymous overlay multicast, Journal of Parallel and Distributed Computing 66 (9) (2006) 1205–1216.



Hyoungshick Kim is a Ph.D. candidate in the Computer Laboratory at the University of Cambridge as a PhD student. He received the B.S. degree from the Department of Information Engineering at Sungkyunkwan University in Korea and M.S. degree from the Department of Computer Science at KAIST in Korea, in 1999 in 2001, respectively. He previously worked for Samsung Electronics as a senior engineer from May 2004 to September 2008. He also served a member of DLNA and Coral standardization for DRM interoperability in home networks. His current research interest is focused on privacy and anonymity in complex networks and distributed systems.

Jaehoon Jeong is currently a software engineer in Brocade Communications Systems. He received the Ph.D. degree under Professor David H.C. Du and Professor Tian He from the Department of Computer Science and Engineering at the University of Minnesota. He received the B.S. degree from the Department of Information Engineering at Sungkyunkwan University in Korea and the M.S. degree from the School of Computer Science and Engineering at Seoul National University in Korea, in 1999 and 2001, respectively. Also, as a researcher in Electronics and Telecommunications Research Institute (ETRI), he has participated in the Internet Standardization in the Internet Engineering Task Force (IETF), such as IPv6 DNS Configuration. He has published three IETF standards of RFC 4339, RFC 5006, and RFC 6106 for IPv6 DNS Configuration. His current research interests are the wireless sensor networking for road networks and the data forwarding in vehicular networks. He is a member of the IEEE and the ACM.