k-Anonymous Message Transmission

Luis von Ahn   Andrew Bortz   Nicholas J. Hopper
Carnegie Mellon University
{lav,abortz,nhopper}@andrew.cmu.edu

May 15, 2003

Abstract

Informally, a communication protocol is *sender k-anonymous* if it can guarantee that an adversary, trying to determine the sender of a particular message, can only narrow down its search to a set of k suspects. *Receiver k-anonymity* places a similar guarantee on the receiver: an attacker, by simply looking at the communication patterns, cannot determine the intended recipient of a message with probability greater than $1/k$. In this paper we introduce the notions of sender and receiver k-anonymity and consider their applications. We show that there exist *simple* and *efficient* protocols which are k-anonymous for both the sender and the receiver in a model where a polynomial time adversary can see all traffic in the network and can control up to a constant fraction of the participants. Our protocol is provably secure, practical, and does not require the existence of trusted third parties. This stands in contrast to the fully anonymous case, where all provably secure protocols in such a strong adversarial model are prohibitively inefficient.

1 Introduction

Anonymous or untraceable communication has been studied extensively in the scientific literature (e.g. [3, 4, 16, 19]). The problem is keeping secret who communicates with whom, as in the case of letters from a secret admirer. The adversary, trying to determine the sender or recipient of a message, is allowed to see all the communications in the network (so a protocol for anonymous communication lets Bob send a secret love letter to the network administrator herself). If used in practice, anonymous communication would have many important applications, such as guaranteeing anonymous crime tip hotlines or allowing "whistle blowers" inside corrupt organizations to leak secrets to the press.

The goal is usually to guarantee full anonymity: an adversary looking at the communication patterns should not learn *anything* about the origin or destination of a particular message. To gain efficiency we concentrate on a weaker goal, k-anonymity: the adversary is able to learn something about the origin or destination of a particular message, but cannot narrow down its search to a set of less than k participants. In other words, k-anonymity guarantees that in a network with n honest participants, the adversary is not able to guess the sender or recipient of a particular message with probability non-negligibly greater than $1/k$, where k is a constant smaller than, but otherwise not related to n. We show that, in our adversarial model, there exists a k-anonymous communication protocol that is far simpler and more efficient than any known fully anonymous communication protocol.

We stress that k-anonymity is sufficient for a variety of applications. In the United States legal system, for example, 2-anonymity would be enough to cast "reasonable doubt," thus invalidating a criminal charge, while 3-anonymity would be enough to invalidate a civil charge (in the absence of other evidence). This is especially relevant after a federal judge in the United States ordered Verizon Communications (a large ISP) to disclose the identity of an alleged peer-to-peer music pirate — a legal decision that could make it easier for the music industry to crack down on file swapping. If the participants in the peer-to-peer network were communicating k-anonymously, the music industry could not prosecute individuals in this manner.[1] k-anonymity is also enough for the protection of privacy in every day transactions, as it effectively breaks data profiling techniques.[2]

The protocol presented in this paper is extremely efficient and provably secure in a strong adversarial model: we assume that the adversary can see all the communications between the participants and can in fact own a significant (smaller than 1/2) fraction of the participants. Participants owned by the adversary can act arbitrarily and attempt to ruin the communications protocol in any possible way — i.e., the adversary not only tries to determine the sender or recipient of particular messages, but also tries to render the anonymous communication protocol useless. As a technical remark, we assume the adversary is computationally bounded (polynomial time) and non-adaptive (in the sense that the adversary must choose which participants to corrupt before the execution of the protocol).

We assume that the network is not adversarially unreliable: messages between communicating parties are always delivered.[3] This assumption is mostly for simplicity, and we stress that our protocol can be used on top of schemes that guarantee reliable communication in an adversarially unreliable setting (e.g. [7]) at the expense of efficiency.

Related Work

For the most part, the study of anonymous communication has focused on efficiency rather than on provable security, and many of the current systems fail when confronted by sufficiently powerful adversaries [20]. Our protocol is provably secure in a strong adversarial model, and achieves efficiency by providing a weaker guarantee than the usual one. It is important to mention that, while k-anonymity is sufficient in many settings, there are cases where full anonymity is is required (e.g. ransom notes).

Below we describe a few of the most influential solutions to the anonymous communication problem and compare them to our proposal.[4]

DC-Nets [3, 19]. DC-Nets is an anonymous broadcast protocol that bases its anonymity on the strength of a secure multiparty sum computation. In this fashion, it is one of the few systems that provides provable security in the absence of trusted parties. Although the original system by Chaum [3] was susceptible to certain attacks, a later variant by Waidner [19] provides an elaborate system of traps and commitments that guarantees robustness and anonymity. However, the poor scalability of DC-Nets makes it unsuitable for medium or large-scale use. In particular, in a network of n users, DC-Nets incurs a cost of $\Omega(n^2)$ protocol messages per anonymous message in every case. Our protocol is similar to DC-Nets, but with a much simpler and efficient method of guaranteeing robustness, better scaling properties, and the possibility to amortize message com-

[1]Though anonymous communication has negative applications such as the distribution of copyrighted material, the *study* of anonymous communication is always a worthwhile endeavor, as it is better to know what can be done and act accordingly than to not know and wait for unethical individuals to discover the results themselves.

[2]The concept of k-anonymity in fact comes from the privacy literature [18].

[3]As long as the network is not adversarially unreliable, protocols such as TCP suffice to provide reliable delivery.

[4]This section is only meant to provide a sample of the previous work so as to put our proposal in context; it is not meant to provide a complete description of the literature. See [9] for a more thorough listing.

plexity over several anonymous messages. Our adversarial model is similar to that assumed in the DC-Nets literature except that we restrict the adversary to run in polynomial time.

Mix-Nets [4] and Onion Routing. Mix-Nets, introduced by David Chaum in 1981, was one of the first concepts for anonymizing communication. The idea is that a trusted "Mix" shuffles messages and routes them, thus confusing traffic analysis. Chaining Mixes together to form a path, combined with Mix-to-Mix (Onion Routing) and end-to-end encryption, offers a form of provable security against a completely passive adversary [4]. Mix-Nets requires the existence of semi-trusted nodes: security is guaranteed as long as one Mix (out of a small constant number of them) is honest.

In all of the Mix-Nets proposals, an active adversary who participates in the system is able to degrade the anonymity of selected messages and users with non-negligible probability [14], and also degrade efficiency through excessive, anonymous usage of its capabilities [12] and selective, undetectable non-participation [20].

Compared to Mix-Nets protocols, our solution incurs fewer network latencies, requires no special trusted nodes, and is provably secure against non-participating active adversaries.

Crowds [16]. Similar to Mix-Nets, Crowds provides paths to disguise the originator of a message. Unlike Mix-Nets, however, paths in Crowds are determined randomly by the machines through which a message passes. Crowds provides sender *probable innocence* against an adversary who controls a certain fraction of the participants (a protocol provides sender probable innocence if the receiver cannot identify the sender with probability greater than $1/2$). However, Crowds provides no protection against a global eavesdropper. k-anonymity can be seen as a further refinement of probable innocence and in particular, our protocol for the case of 2-anonymity is competitive with Crowds in terms of round complexity, slightly worse in communication complexity and incurs much heavier computational costs, while providing provable security in a stronger adversarial model.

CliqueNet [17]. CliqueNet combines small DC-Nets with a routing layer to mitigate the problems of DC-Nets scalability while also preserving some of its anonymity guarantees. CliqueNet has the undesirable feature, however, that an adversary who controls ℓ network nodes can completely compromise the anonymity of $\ell - 1$ other nodes of his choice. Furthermore, CliqueNet's routing layer induces a high amount of unnecessary network latency, and is not secure against non-participation, allowing an adversary who controls a few nodes to partition the network. Our protocol is similar to CliqueNet in that we also divide the network into small DC-Nets-like components, but different in that we provide provable security against strong adversaries.

Organization of the Paper

Section 2 presents the basic cryptographic notions and definitions we will need for the paper. Section 3 introduces the definitions for k-anonymous communication, Section 4 introduces the novel protocol that achieves k-anonymity for both the sender and the receiver, and Section 5 delineates how to construct a communications network that can guarantee k-anonymity. Finally, Section 6 concludes with a discussion and some open questions.

2 Preliminaries

Notation

A function $\mu : \mathbb{N} \to [0,1]$ is said to be *negligible* if for every $c > 0$, for all sufficiently large n, $\mu(n) < 1/n^c$. Let S be a set, then $x \leftarrow S$ denotes the action of choosing x uniformly from S. U_k denotes the set of k-bit strings. We denote the set of integers $\{1, \ldots, n\}$ by $[n]$. We will use $\mathbb{Z}_m$ to denote the additive group of integers modulo m, and $\mathbb{Z}_m^*$ to denote the multiplicative group of integers modulo m. When we say *split* $x \in \mathbb{Z}_m$ *into* n *random shares* $s_1, \ldots, s_n$ we mean choose $s_1, \ldots s_{n-1}$ uniformly at random from $\mathbb{Z}_m$ and set $s_n = x - (s_1 + \cdots + s_{n-1}) \bmod m$.

The Model

We assume a network of n parties $\{P_1, \ldots, P_n\}$, of which a fraction β are controlled by a non-adaptive polynomial time adversary, who may also monitor the communications between all parties. Parties under the control of the adversary may behave arbitrarily, while the remaining *honest* parties are constrained by the protocol. We assume that the network is reliable: messages between parties are always delivered. We also assume the existence of a public-key infrastructure which allows secure authenticated channels between all pairs of parties.

Pedersen Commitments

Let p and q be primes such that q divides $p-1$, and let $g, h \in \mathbb{Z}_p^*$ have order q. (It is easy to see that both g and h generate the unique subgroup of order q in $\mathbb{Z}_p^*$.) The following commitment scheme is due to Pedersen [13], and is based on the difficulty of finding $\log_g(h)$ (all the multiplications are over $\mathbb{Z}_p^*$):

> **Commit Phase:** to commit to $s \in \mathbb{Z}_q$, choose r uniformly from $\mathbb{Z}_q$ and output $C_r(s) = g^s h^r$.
>
> **Reveal Phase:** To open the commitment, simply reveal s and r.

For any s, the commitment $C_r(s) = g^s h^r$ is uniformly distributed over the unique subgroup of order q in $\mathbb{Z}_p^*$, so that $C_r(s)$ reveals no information about s. Furthermore, the committer cannot open a commitment to s as $s' \neq s$ unless she can find $\log_g(h)$.

Secure Multiparty Addition

A secure multiparty addition protocol allows parties $P_1, \ldots, P_n$, each with a private input $X_i \in \mathbb{Z}_m$, to compute $X_1 + \ldots + X_n$ in such a way that P_i, regardless of its behavior, learns nothing about X_j (except what can be derived from $X_1 + \ldots + X_n$) for any P_j which conforms to the protocol $(i \neq j)$. The following commonly-known scheme implements secure multiparty addition: each party P_i splits X_i into n random shares $s_{i,1}, \ldots, s_{i,n}$ such that $\sum_j s_{i,j} = X_i$ and sends share $s_{i,j}$ to party j; later all parties add every share that they have received and broadcast the result. It is easy to see that the sum of all broadcasts equals $X_1 + \ldots + X_n$, and that it is impossible for party j to learn anything about X_i (for $i \neq j$).

For the rest of this paper, we use the following (also well-known) modification of the above scheme (the commitments are used in order to ensure that all parties adhere to the protocol; e.g., parties shouldn't be able to cheat by sending inconsistent shares):

Commitment Phase: P_i uniformly chooses shares $s_{i,1}, \ldots, s_{i,n}$ subject to $\sum_j s_{i,j} = X_i$, and computes commitments $C_{i,j} = C_{r_{i,j}}(s_{i,j})$. P_i then broadcasts the ordered commitments $\{C_{i,j} : 1 \leq j \leq n\}$ to all parties.

Sharing Phase: For each $j \neq i$, P_i sends to P_j the values $r_{i,j}, s_{i,j}$. P_j checks that $C_{r_{i,j}}(s_{i,j}) = C_{i,j}$.

Broadcast Phase: P_i computes the values $B_i = \sum_j s_{j,i}$ and $R_i = \sum_j r_{j,i}$ and broadcasts B_i, R_i. All players check that $C_{R_i}(B_i) = \prod_j C_{j,i}$.

Result: Each player computes the result as $X = \sum_i B_i$, computes $R = \sum_i R_i$ and checks that $C_R(X) = \prod_{i,j} C_{i,j}$.

Secure multiparty addition and anonymous communication are related (an observation which seems to be due to David Chaum and forms the basis of DC-NETS), in that a protocol for secure multiparty addition can be used to perform anonymous broadcast. Assume that at time t party j wants to broadcast the message $X_j \neq 0$ anonymously, while the other parties do not wish to broadcast anything; then by performing a multiparty addition with $X_i = 0$ (for $i \neq j$), all the parties learn $X_1 + \ldots + X_n = X_j$, but nobody learns where X_j came from. If more than one party tries to transmit at the same time, however, a collision occurs and the parties have to wait until the next time step. For this reason DC-NETS use a complicated reservation mechanism to keep the adversary from jamming the channel: if the adversary controls a single participant, it can simply send a message at every time step. Our protocol is based on secure multiparty sum computations, and one of the novel aspects of our work is the simple mechanism that we use to prevent the adversary from jamming the channel.

Zero-Knowledge Proofs

We will use a zero-knowledge proof [11] to enforce the robustness of our protocol. Informally, a zero-knowledge proof is a protocol which allows a *prover* program P to convince a *verifier* program V of the veracity of a statement while giving the verifier no additional information. In this paper we will only require security in the case of an *honest verifier* (i.e., the verifier follows the program V). We stress that there exist standard techniques ([10], [2]) to convert the particular type of honest-verifier zero-knowledge proof that we will use into a proof which is secure even against a dishonest verifier.

Definition 1. *A protocol* (P, V) *is* honest verifier zero-knowledge *if there is an efficient program* S *(a* simulator*) such that the output of* $S(x)$ *and the view of* V *upon interaction with* $P(x)$ *are indistinguishable.*

An example is the following protocol for proving *knowledge* of the discrete logarithm of $x = h^r \bmod p$ (where $p = 2q+1$ and p, q are prime) originally due to Chaum *et al.* [5]:

1. P picks $\sigma \leftarrow \mathbb{Z}_q$
$$P \longrightarrow V \ : \ y = h^\sigma \bmod p$$
2. $V \longrightarrow P \ : \ z \leftarrow \mathbb{Z}_q$
3. $P \longrightarrow V \ : \ w = rz + \sigma \bmod q$
4. V accepts if $x^z y = h^w$

The honest-verifier simulator for this protocol first selects the values $z, w \leftarrow \mathbb{Z}_q$ and sets $y = h^w/x^z \bmod p$, then outputs the conversation y, z, w. A prover can cheat in this protocol only with very small probability, $1/q$.

3 Anonymous Message Transmission

Anonymous message transmission is a computation in which each party P_i has as input a pair $(msg_i, p_i) \in (\mathcal{M} \times [n]) \cup \{(\texttt{nil}, \texttt{nil})\}$. Intuitively, at the end of a round each P_i should learn the set of messages msg_j with $p_j = i$, but not the identity of the senders.

We let $H \subset \{P_1, \ldots, P_n\}$ denote the set of honest parties. We denote by $\mathcal{P}(P_1(msg_1, p_1), \ldots, P_n(msg_n, p_n))$ the random variable distributed according to the adversary's *view* of the protocol $\mathcal{P}$ when each P_i has input (msg_i, p_i). We denote by $\mathcal{P}(P_i(msg_i, p_i), *)$ the adversary's view of $\mathcal{P}$ when P_i has input (msg_i, p_i) and the other inputs are set arbitrarily.

Definition 2. *A protocol is* sender anonymous *if for every pair $P_i, P_j \in H$, and every pair $(msg, p) \in (\mathcal{M} \times [n]) \cup \{(\texttt{nil}, \texttt{nil})\}$, $\mathcal{P}(P_i(msg, p), *)$ and $\mathcal{P}(P_j(msg, p), *)$ are computationally indistinguishable.*

That is, a protocol is sender anonymous if the adversary may not distinguish between any of the honest parties as the sender of a message, regardless of who the receiver is; i.e., the adversary "gains no information" about the sender.

Definition 3. *A protocol is* receiver anonymous *if for every $P' \in H$, for every $msg \in \mathcal{M}$ and every $P_i, P_j \in H$, $\mathcal{P}(P'(msg, P_i), *)$ and $\mathcal{P}(P'(msg, P_j), *)$ are computationally indistinguishable.*

According to the previous definitions, the trivial protocol in which no party transmits anything is both sender and receiver anonymous. Non-triviality is captured by Definition 6 below.

Assuming that the protocol is non-trivial (i.e., useful), sender anonymity requires every honest party to send at least one *protocol* message per anonymous message delivered, even if they have no message as an input. Thus any protocol which is sender anonymous has a worst-case lower bound of n protocol messages per input message. If n is large, this lower bound makes it unlikely that a system providing full anonymity can be fielded in practice.

Definition 4. *A protocol $\mathcal{P}$ is* sender k-anonymous *if it induces a partition $\{V_1, \ldots, V_l\}$ of H such that:*

1. *$|V_s| \geq k$ for all $1 \leq s \leq l$; and*
2. *For all $P_i, P_j \in V_s$, for every $(msg, p) \in (\mathcal{M} \times [n]) \cup \{(\texttt{nil}, \texttt{nil})\}$, $\mathcal{P}(P_i(msg, p), *)$ and $\mathcal{P}(P_j(msg, p), *)$ are computationally indistinguishable.*

That is, each honest party's messages are indistinguishable from those sent by at least $k - 1$ other honest parties.

Definition 5. *A protocol $\mathcal{P}$ is* receiver k-anonymous *if it induces a partition $\{V_1, \ldots, V_l\}$ of H such that:*

1. *$|V_s| \geq k$ for all $1 \leq s \leq l$; and*
2. *For all $P_i, P_j \in V_s$, for every $P' \in H$, $msg \in \mathcal{M}$, $\mathcal{P}(P'(msg, P_i), *)$ and $\mathcal{P}(P'(msg, P_j), *)$ are computationally indistinguishable.*

That is, each message sent to an honest party has at least k indistinguishable recipients.

In addition to the anonymity guarantees, we will require that the communications protocol be robust against an adversary trying to render it useless. We capture this intuition with the notion of *robustness.*

Definition 6. *Let $\alpha \in [0, 1]$. A protocol $\mathcal{P}$ is α-*robust *if either one of the two following conditions is true:*

1. *(Fairness) For all $P' \in H$ and for all $(msg, P_i) \in (\mathcal{M} \times [n])$, the probability (over the randomness of P') that party P_i receives msg is at least α.*
2. *(Detection) The set S of parties who deviate from $\mathcal{P}$ is non-empty and there is a single $P_i \in S$ such that for all P_j, if $P_j \notin S$, P_j outputs P_i.*

Note that in case the execution is not fair and multiple parties deviate from $\mathcal{P}$, we only require that a single misbehaving party be caught.

4 Transmission Protocol

Our solution to the k-anonymous message transmission problem is similar to Chaum's [3] DC-Nets but features two important innovations.

First, we randomly partition the n parties into smaller groups of size $M = k/(1 - 2\beta)$ (recall that β is the fraction of the parties that the protocol tolerates the adversary to control) so that with high probability k members of each group are honest. Each group performs essentially the multiparty sum protocol described in Section 2, where the input X_i is expressed as a (msg, g) pair describing the message msg to be transmitted and the group g of the receiver. This guarantees sender k-anonymity (because with high probability $k - 1$ other members of the group are honest) as well as receiver k-anonymity (because each message is received by $M \geq k$ participants).

Second, each group runs $2M$ copies of the multiparty sum protocol in parallel, allowing each party to transmit in at most one parallel copy (this is to achieve robustness). We give a protocol which allows the detection of at least one non-conforming party in each round where access to this shared channel was not fair. Since each group has only $O(k)$ non-conforming parties, an adversary can only cause $O(k)$ protocol failures in each group; and no protocol failure compromises the anonymity of any honest party. In comparison, previous solutions built around DC-Nets allow $O(n^2)$ protocol failures, and exposing a cheater in those systems may involve compromising the anonymity of a message.

Protocol 1. *k-AMT.*

Precondition: Assume that the n parties are partitioned into groups of size M, with each group having at least k honest participants (in Section 5 we discuss how this precondition is met). Below are the instructions to be performed by each group individually. For notational simplicity, we denote the parties in the current group by $P_1, ..., P_M$.

Input: Each party P_i in the group has input g_i, the group the receiver belongs to, and msg_i, a message. (msg_i, g_i) will be interpreted as an element of $\mathbb{Z}_q$, where q is a large prime that divides $p - 1$ (p is also a prime). We identify $(msg_i, g_i) = (\texttt{nil}, \texttt{nil})$, indicating "no message this round," with $0 \in \mathbb{Z}_q$.

Commitment Phase: To be performed by all participants in the group (i.e., $1 \leq i \leq M$):

1. P_i chooses $l \leftarrow \{1, ..., 2M\}$ and sets $X_i[l] = (msg_i, g_i)$ and $X_i[t] = 0$ for $1 \leq t \neq l \leq 2M$.
2. P_i splits $X_i[t] \in \mathbb{Z}_q$ into M random shares $s_{i,1}[t], \ldots, s_{i,M}[t]$.
3. P_i chooses $r_{i,j}[t] \leftarrow \mathbb{Z}_q$, for all $1 \leq j \leq M, 1 \leq t \leq 2M$.
4. P_i computes commitments $C_{i,j}[t] = g^{s_{i,j}[t]} h^{r_{i,j}[t]} \bmod p$.
5. P_i broadcasts the commitments $C_{i,j}[t]$, $1 \leq j \leq M, 1 \leq t \leq 2M$, to all members of the group.

Sharing Phase: For each j $(1 \leq j \leq M)$:

1. P_i sends P_j the ordered values $\{s_{i,j}[t], r_{i,j}[t] : 1 \leq t \leq 2M\}$.
2. P_j checks that
$$C_{i,j}[t] = g^{s_{i,j}[t]} h^{r_{i,j}[t]} \bmod p$$
for each $1 \leq t \leq 2M$.

Local Broadcast Phase:

1. P_i calculates $B_i[t] = \sum_j s_{j,i}[t] \bmod q$ and $R_i[t] = \sum_j r_{j,i}[t] \bmod q$ and broadcasts the ordered values $\{B_i[t], R_i[t] : 1 \leq t \leq 2M\}$ to all members of the group.
2. P_j checks that
$$g^{B_i[t]} h^{R_i[t]} = \prod_j C_{j,i}[t] \bmod p \ ,$$
for each $1 \leq t \leq 2M, 1 \leq i \leq M$.

Transmission Phase: To be performed by all participants in the group (i.e., $1 \leq i \leq M$):

1. For all t $(1 \leq t \leq 2M)$, P_i calculates $X[t] = (Msg[t], G[t]) = \sum_i B_i[t] \bmod q$.
2. For all t $(1 \leq t \leq 2M)$, if $X[t] \neq 0$, P_i sends $Msg[t]$ to every member of group $G[t]$.

Robustness

Suppose at the conclusion of the transmission phase, at most M of the $2M$ values $X[t]$ were non-zero. Then this execution was *fair*: each P_i had probability at least $\frac{1}{2}$, over its own choices, of successfully transmitting msg_i. On the other hand, if *more* than M of the $X[t]$ were non-zero, then at least one P_i had more than one $X_i[t] \neq 0$. We now describe an *honest verifier* zero-knowledge proof that allows each honest party to prove that they set at most one $X_i[t]$ to a non-zero value, assuming it is hard to compute $\log_g(h)$ (this allows the honest players to identify at least one party P_i with more than one $X_i[t]$ not equal to zero).

Intuitively, this protocol uses the well-known "cut-and-choose" technique: player P_i prepares new commitments $C_i'[t]$ to the values $X_i[t]$ and randomly permutes them. Then the verifier may choose either to have the prover open $2M - 1$ of the (permuted) $C_i'[t]$ values to zero, or to have the prover reveal the permutation and prove (in zero-knowledge) that he can open the commitments $C_i'[t]$ and $C_i[t]$ (for each $1 \leq t \leq 2M$) to the same value.

Protocol 2. *Zero-Knowledge proof that at most one* $X_i[t] \neq 0 \bmod q$.

1. P_i chooses $r'[t] \leftarrow \mathbb{Z}_q, 1 \le t \le 2M$, and $\pi \leftarrow \mathbb{S}_{2M}$ (a permutation on $\{1, \dots, 2M\}$). Define

$$\begin{aligned} \rho_i[t] &= \sum_j r_{i,j}[t] \ , \\ C_i[t] &= \prod_j C_{i,j}[t] \bmod p = C_{\rho_i[t]}(X_i[t]) \ , \\ \rho'_i[t] &= r_i[t] + r'[t] \bmod q \ , \\ C'_i[t] &= C_i[t] h^{r'[t]} \bmod p = C_{\rho'_i[t]}(X_i[t]) \ . \end{aligned}$$

   $P_i \longrightarrow V \ : \ \langle \kappa[t] = C'_i[\pi(t)] \rangle_{t=1,\dots,2M}$.

2. $V \longrightarrow P_i : b \leftarrow \{0, 1\}$

3. If $b = 0$, then:

   (a) P_i sets l such that $X_i[l] \neq 0$ if l exists, or chooses $l \leftarrow \{1, \dots, 2M\}$ otherwise.

   $$P_i \longrightarrow V : \langle \xi[t] = \rho'_i[\pi(t)] \rangle_{\pi(t) \neq l} \ .$$

   (b) V accepts iff $h^{\xi[t]} = \kappa[t] \bmod p$ for all $t \neq \pi^{-1}(l)$.

   Otherwise, P_i proves that C' is a commitment to a permutation of C by revealing π and proving knowledge of the discrete log of $x[t] = C'_i[t]/C_i[t] = \kappa[\pi^{-1}(t)]/C_i[t]$:

   (a) P_i picks values $\sigma[t] \leftarrow \mathbb{Z}_q$

   $$P_i \longrightarrow V \ : \ \pi, \langle y[t] = h^{\sigma[t]} \bmod p \rangle_t$$

   (b) $V \longrightarrow P_i \ : \ \langle z[t] \leftarrow \mathbb{Z}_q \rangle_t$

   (c) $P_i \longrightarrow V \ : \ \langle w[t] = r'[t] z[t] + \sigma[t] \bmod q \rangle_t$

   (d) V accepts if $x[t]^{z[t]} y[t] = h^{w[t]}, 1 \le t \le 2M$

The above protocol is public-coin, honest-verifier statistical zero knowledge. In practice, we may implement the verifier by calls to a cryptographic hash function and obtain security in the Random Oracle Model [2], or the verifier may be implemented by the remaining parties through a subprotocol in which each party non-malleably commits to her random bit(s) and then reveals the bits; the randomness used is then the exclusive-or of each party's random string. So long as there is one honest verifier this approach will work; a party which refuses to participate in this subprotocol can be recognized as the cheating party, fulfilling the detection criterion.

Notice that this protocol is very efficient: with security parameter λ, (the number of parallel repetitions of Protocol 2) the number of rounds is constant, the total number of bits transmitted is $O(M\lambda \lg p) = O(k\lambda \lg p)$, and a non-conforming party is caught with probability at least $1 - 2^{-\lambda}$. However, even if the protocol were less efficient, since the protocol need only be executed when cheating takes place, and all cheaters can be caught with high probability, the cost of detection when amortized over many rounds is essentially zero.

We stress that situations in which a dishonest party does not comply with the protocol by sending incorrect messages or different messages to different parties can be dealt with by requiring signatures on all messages and using a simple protocol where all parties reveal all signed messages received; any case in which two different messages signed by the same party appear reveals a dishonest party.

Security

Theorem 1. *If group G has at least k honest parties, then Protocol k-AMT is sender k-anonymous for senders in group G.*

Proof. (Sketch) In each parallel round, the multiparty sum protocol guarantees that no adversary may determine the inputs of any honest parties; thus the adversary may not distinguish between the case that $X_i[t] = 0$ and $X_i[t] = Msg[t]$ for any honest party. $\square$

Theorem 2. *If every group G has at least k honest parties, then Protocol k-AMT is receiver k-anonymous.*

Proof. (Sketch) Each message sent to an honest party P_i is received by all parties in P_i's group; since there are at least k honest parties in this group, the adversary cannot distinguish between these parties as the recipients. $\square$

Theorem 3. *Protocol 2 is sound: if for some i, there exist $t \neq t'$ such that $X_i[t] \neq 0$ and $X_i[t'] \neq 0$ then $|\Pr[V \text{ accepts}] - \frac{1}{2}|$ is negligible.*

Proof. (Sketch) Suppose V chooses $b = 0$; then, if the commitments C'_i are formed correctly P_i must compute $\log_g h \pmod p$ in order to open one of $C'_i[t], C'_i[t']$ to zero. If computing discrete logarithms modulo p is hard, then this happens with negligible probability. Likewise, if V chooses $b = 1$, then if the commitments C'_i are malformed, P_i must compute $\log_h g$ in order to make V accept (by the soundness of the discrete logarithm subprotocol in step 3). So for the honest V, regardless of the formation of the commitments C'_i, P_i has probability at most 1/2 plus a negligible factor of convincing V to accept. $\square$

Theorem 4. *Protocol 2 is honest-verifier zero-knowledge.*

Proof. (Sketch) We exhibit a simulator for the honest-verifier case: flip a coin representing the bit b in step 2. If $b = 0$, form the commitments $C'_i[t] = C_{r'[t]}(0)$ from step 1, choose a random $l \in 1, \ldots, 2M$ and reveal $r'[1], \ldots, r'[l-1], r'[l+1], \ldots, r'[2M]$ in step 3. If $b = 1$, form the commitments $C'_i[t]$ in the same manner as the honest prover, and use the honest-verifier simulator for the discrete logarithm protocol in step 3. $\square$

Theorem 5. *If the precondition for Protocol 1 holds, Protocol 1 and Protocol 2 together give a $\frac{1}{2}$-robust k-anonymous transmission protocol.*

Efficiency

Because we detect cheaters with high probability, we may consider the typical case to be that where all participants follow the protocol exactly. In this case, the round complexity is 4. In terms of message complexity, in every case we transmit $O(M^2)$ messages for every anonymous message sent. The bit complexity per anonymous bit sent is $O(M^3)$ in the worst case.

In the case where on average, $O(M)$ parties send anonymous messages per round, the Transmission Phase of Protocol 1 still transmits $O(M^2)$ protocol messages for every anonymous message sent. However, there are alternate strategies that allow amortizing this message complexity over the anonymous messages of the group.

One alternative is to replace this transmission phase by another in which, for each t such that $X[t] \neq 0$, each P_i randomly chooses $\lceil \frac{c}{1-2\beta} \rceil$ members of $G[t]$ and sends $Msg[t]$ to those parties. In this case, when $O(M)$ parties transmit anonymously the ratio of protocol messages to anonymous messages is $O(M)$, and the ratio of protocol bits to anonymous bits is $O(M^2)$. However, with probability e^{-c} all of the honest parties of the sending group fail to send to the intended recipient of $Msg[t]$ in this transmission phase, and this condition is undetectable by the anonymous sender, requiring forward erasure correction over message blocks.

Another alternative trades round complexity for message complexity in the "best case:" After each P_i in the sending group computes $X[t]$, P_i sends $Msg[t]$ to the i^{th} member Q_i of $G[t]$. Each Q_i then sends P_i a signature on $Msg[t]$. Finally each P_i collects all such signatures and broadcasts these signatures to the other members of his group. In this alternative, the round complexity is 6, but again when $O(M)$ anonymous messages are transmitted the ratio of protocol messages to anonymous messages is $O(M)$ and the ratio of protocol bits to anonymous bits is $O(M^2)$; and any member of the sending group who fails to forward anonymous messages is caught.

We intend for our protocol to be used over the Internet or networks of similar characteristics. Since in such networks throughput is frequently constrained by network latency, our protocol is particularly efficient, though at the expense of increased computational and message complexity.

5 Network Construction

The protocols in the previous section work for any network which has already constructed the individual groups. Here we present several strategies related to the efficient, scalable construction and management of this group structure.

Group Formation and Management

We propose that a simple protocol be used to construct the groups. The formation of groups should be such that parties cannot choose which group they belong to. In an initialization phase, interested parties may securely construct the list $\langle P_1, \ldots, P_n \rangle$ either through a small group of trusted registration servers or through a secure group membership protocol such as that of [15]. The parties then choose a session identity S, for example, using a cryptographic hash function H applied to the initial parameters of the network. The number of groups is determined as the largest power of 2 smaller than $n(1-2\beta)/2k$, say 2^m. Then each P_i determines his group number by the m least significant bits of $H(S||P_i)$; thus any party, given the list of participants, can determine the group of any other party, and the other participants in his own group.

Minimizing Turnover

If a significant number of honest parties leave the network (even temporarily) then the k-anonymity property may sometimes be violated. A possible approach to minimize this risk is by charging a high computational cost to rejoin a group, using a protocol such as Dwork and Naor's moderately hard functions [8] or Back's Hashcash [1].

Rate Adjustment

Notice that a significant barrier to the implementation of a fully anonymous protocol such as DC-NETS is the need to fully synchronize n hosts when n is large. In the protocol proposed here, there is no such requirement — the groups may operate asynchronously of one another. Because of that, each individual group may optimize its time between rounds to approximate the average sending rate of the group. This can be accomplished automatically using the fact that the outcome of the protocol gives a good estimate of the number of parties transmitting each round; so if no parties transmit, an additive increase in the intra-round gap may be used, and if many parties transmit, a multiplicative decrease may be used, as in other fair communications protocols.

6 Conclusions

In this paper, we have introduced the notion of k-anonymous message transmission, by analogy to the concept of k-anonymity from the privacy literature. Using this notion, we are able to give simple and efficient protocols for anonymous message transmission which have provable security against a very strong adversary. We believe an interesting avenue for further research is to investigate whether other multiparty computation tasks can also be simplified using a similar approach, i.e. by weakening the security goals in a manner which is still sufficient for many applications. We also believe an important future step is the implementation of our protocol in order to determine the actual overhead introduced and the achievable throughput.

References

[1] Adam Back. Hashcash. Unpublished manuscript, May 1997. Available electronically at `http://www.cypherspace.org/hashcash/`.

[2] Mihir Bellare and Phil Rogaway. Random Oracles are Practical. *Computer and Communications Security: Proceedings of ACM CCS*, pp 62-73, 1993.

[3] David Chaum. The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. *Journal of Cryptology* 1(1), pp 65-75, 1988.

[4] David Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM* 24(2), pp 84-88, 1981.

[5] David Chaum, Jan-Hendrik Evertse, Jeroen van de Graaf and René Peralta. Demonstrating Possession of a Discrete Logarithm Without Revealing It. Advances in Cryptology: CRYPTO 86, pp 200-212, 1987.

[6] Danny Dolev, Cynthia Dwork, Moni Naor. Non-Malleable Cryptography (Extended Abstract). STOC 1991, pp 542-552.

[7] Danny Dolev, Cynthia Dwork, Orli Waarts and Moti Yung. Perfectly Secure Message Transmission. J. ACM 40(1), pp 17-47, 1993.

[8] Cynthia Dwork and Moni Naor. Pricing via Processing, or: Combating Junk Mail. Advances in Crypology: CRYPTO 92, pp 139-147, 1993.

[9] The GNUnet website. `http://www.ovmj.org/GNUnet/`.

[10] Oded Goldreich, Amit Sahai, and Salil Vadhan. Honest Verifier Statistical Zero-Knowledge Equals General Statistical Zero-Knowledge. In *Proceedings of 30th Annual ACM Symposium on Theory of Computing*, pp. 399-408. May 1998.

[11] Shafi Goldwasser, Silvio Micali, Charles Rackoff. The Knowledge Complexity of Interactive Proof-Systems (Extended Abstract). STOC 1985, pages 291-304.

[12] David Mazieres and M. Frans Kaashoek. The Design, Implementation, and Operation of an Email Pseudonym Server. *CCS 98: Proceedings of the 5th ACM Conference on Computer and Communications Security*, pp 27-36, 1998.

[13] Torben P. Pedersen. Non-Interactive and Information Theoretic Secure Verifiable Secret Sharing. *Advances in Cryptology, CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pp 129-140. Santa Barbara, CA, 1991.

[14] Andreas Pfitzmann and Michael Waidner. Networks Without User Observability – design options. *Proceedings of Eurocrypt 1985*, pp 245-253.

[15] Michael K. Reiter. A secure group membership protocol. *IEEE Trans. Software Engineering* 22(1), pp 31-42, 1996.

[16] Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security* 1/1, November 1998, pp 66-92.

[17] Emin Gün Sirer, Milo Polte, and Mark Robson. CliqueNet: A Self-Organizing, Scalable, Peer-to-Peer Anonymous Communication Substrate. Unpublished manuscript, December 2001. Available electronically at `http://www.cs.cornell.edu/People/egs/papers/cliquenet-iptp.pdf`.

[18] Latanya Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10 (5), 2002; pp 557-570

[19] Michael Waidner. Unconditional sender and recipient untraceability in spite of active attacks. *Eurocrypt '89*, volume *Lecture Notes in Computer Science* of 434, pp 302-319. Springer-Verlag, 1989.

[20] M. Wright, M. Adler, B. Levine, and C. Shields. An analysis of the degradation of anonymous protocols. *Proceedings of ISOC Symposium on Network and Distributed System Security*, February 2002.