Provable Anonymity Against Traffic Analysis *

Ron Berman, Amos Fiat, Amnon Ta-Shma

July 18, 2005

Abstract

With the advent of peer to peer networks, anonymity is grasped as a desired property of any well designed system for exchanging information between parties. Previous work dealing with anonymity and privacy is mostly application driven and intuitively based, paying more attention to implementation details than to rigorous security analysis.

This work focuses on unlinkability, which is one of the possible types of anonymity. We study the unlinkability of an efficient variant of David Chaum's classic protocol for anonymous data communication and we prove that it is secure against adaptive adversaries even having prior information. Prior to our work, most attempts to make a rigorous analysis were either inefficient or were based on ad-hoc assumptions.

Our technique heavily relies on information theory. We use information theory both in formalizing the intuitive definitions and in proving the security properties of our protocol.

1 Introduction

Anonymous communication is widely used today, *e.g.*, privately browsing sensitive medical information or anonymously notifying the police of an ongoing crime. The term "anonymity" is usually referred to as the property of being unidentifiable or indistinguishable when being put together with a group of people.

Common methods for achieving anonymity include:

- Introducing ambiguity into the communications pattern. *I.e.*, making communication less specific and explicit.
- Joining a large group of entities and hiding the unique communication pattern within a larger more random one.
- Letting others do the job for us, using a trusted mediator.

*A preliminary version of the paper appeared in the proceedings of Financial Cryptography 2004.

- Using encryption to hide communication contents.

This short list is naturally not complete, and as a matter of fact, not all methods or combination of methods provide the needed security. As surprising as it may be, even the definition of anonymity is unclear in many situations, thus resulting in incomplete work, and more significantly, systems with unproven security properties.

1.1 Informal Definition of Anonymity

Pfitzmann and Köhntopp [PK00] to put some order into the various interpretations of the term "anonymity" and create commonly accepted terms. The goal of anonymity is subdivided into three smaller more specific objectives — sender anonymity, recipient anonymity and unlinkability of sender and receiver:

- An anonymity set of a certain item is the group of items that serve as an obscuring background for a certain property of the designated item. If, for example, the nodes in an anonymity set of a certain node can only send messages, then this designated node can only achieve sender anonymity.
- Unlinkability of two items means that these two items are not more related than before the system was run, given the posteriori knowledge. As unlinkability of two items depends on the relationship possible between those items and the relationship with the rest of the anonymity set, we receive the following subdivision:

  - Sender unlinkability means it is impossible to link a message with the identity of its sender.

  - Receiver unlinkability is the same as the above only the linkage is with the receiver of a message.

  - Relationship unlinkability means it is impossible to link a sender and a receiver as a transaction couple. This term is usually referred to as just "unlinkability".

All of the mentioned types have been under research for quite some time.

It is clear that having more mathematical definitions that quantify anonymity is desirable, and we later on survey previous work and present our definition using the language of information theory. Beyond the clarity of our new definitions, they enable us to define unlinkability in the presence of prior information, an important issue that was ignored in previous work. We also receive an added bonus of using information theory tools.

1.2 Adversary Models

Past work, even when dealing specifically with the subject of anonymous communication, has introduced a great variety of adversary strengths, threat models and attack descriptions. The importance of choosing the correct model as the basis of analysis cannot be stressed enough. We now describe three types of general adversaries which seem to be most natural.

1.2.1 Before we start

Our main focus is unlinkability, and so we (and most of the previous work on unlinkability) make public the list of message senders and receivers at every stage. We stress that if many trials take place, this by itself leaks information. For example, suppose that each user is either bad or good and bad users access other data than good users, then only a few runs of the protocol are necessary to identify a user as either bad or good. A similar result has appeared in [Dan03]. As a result, in this work (and most of previous work) we focus on single runs of the communication protocol.

1.2.2 Passive Adversary

A passive adversary, as its name suggests, cannot initiate any traffic in the network and we can think of it as after-the-event adversary. After the communication protocol is run, the adversary has access to some of the information used on the communication lines, and it tries to infer relationships between parties in the network.

A passive adversary can listen to a communication link between any two nodes, and in particular is able to determine if there was communication on that link at a specific point in time. The adversary can also listen to a node. In such a case, the adversary is able to follow all links connected with the node, and tell whether there was or wasn't any traffic on any such link at any specific time. We stress that we allow all types of adversaries to collude amongst themselves and share any information they acquire.

Passive adversaries have potentially the power of recording and understanding message content as they pass through links under their surveillance (content analysis). Using cryptographic methods allows the circumvention of content analysis, as described in Section 1.5.2. Having that reduction we can assume a passive adversary can only determine if there was or wasn't traffic on some links, and can only perform traffic analysis.

1.2.3 Adaptive Adversary

An adaptive adversary is a passive adversary with the extra ability of initiating a message from a node in the system under its control.

1.2.4 Malicious Adversary

The third and most powerful type of adversary is called malicious. Generally speaking, this type of player has unlimited power within the network part it controls. A malicious player can inject new messages into the system, change messages as they pass through a controlled node or link and delete or delay a message. In this paper we do not deal with malicious adversaries.

1.3 Efficiency

Say we have N users in our system, and at each given time at most M users want to send a message. When measuring efficiency one may consider

- *delay*, also known as parallel time, *i.e.*, the number of rounds a protocol takes, or,
- *load*, *i.e.*, the total number of messages transmitted throughout the protocol, or,
- *both*.
- The term *overhead* is used to denote the load divided by M. This is the true overhead (measured in messages) for a protocol, imposed on each participating node.

Rackoff and Simon use delay as a measure of efficiency. They say a protocol is *efficient* if the maximum number of rounds is $O(\text{polylog}(N))$. However, mix-based protocols in general, and Rackoff and Simon's algorithm in particular, require all honest nodes to participate at each round — irrespective of whether they have anything to send or not. Thus, while the number of rounds is small the total number of messages is very high and the system is highly inefficient in terms of message overhead.

We define a protocol to be efficient if both the delay is $O(\text{polylog}(N))$ and the load is at most $O(M \cdot \text{polylog}(N))$. Thus, the load on the system is proportional to its use.

1.4 Previous work

A summary of the protocols surveyed in this section appears in Table 1.

1.4.1 A trivial inefficient solution using broadcast

If a broadcast mechanism is available in the network, and the broadcast is implemented such that a receiver does not know the origin of a message[1], then a simple approach to unlinkability is to have every node in the network

[1]Not all broadcast mechanisms have this property. However, several types of current technology's LANs posses this ability. Ethernet is arguably one such example.

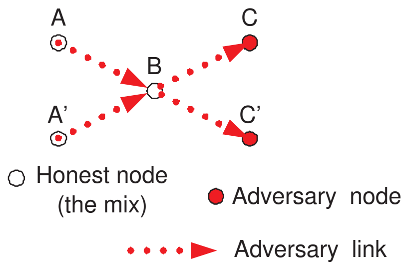

Figure 1: *Chaumian mixes: traffic analysis ambiguity is achieved by honest nodes that collect multiple incoming messages before sending out the "peeled" messages to the next stage.*
"Empty" vertices are honest, while solid vertices are under adversary control. All links are under adversary control.

broadcast every data item it holds to all other nodes continuously. A node interested in a certain item just listens to the network continuously until the item sought for is broadcast. The protocol achieves anonymity against adaptive adversaries, but is inefficient and seems impractical in large networks.

1.4.2 Chaumian Mix-Based Systems

In 1980 David Chaum introduced a novel approach to communications anonymity [Cha81]. Chaum's ideas serve as the basis for most practical implementations of communication systems featuring anonymity. The model assumes nodes hold data items, and all data items are of the same length, or padded to the same length. Some nodes and *all links* are under control of an adversary, others are not and are called *honest*. Chaum's system uses two fundamental building blocks.

A Mix A mix (a.k.a. a Chaumian mix) is a mediating computer system between a sender of a message and its receiver. The purpose of a mix is to introduce ambiguity into traffic patterns, thus disallowing the ability to trace the path of a message from its sender to its receiver.

The mix system accepts batches of messages from senders, each one with its desired target address, and then forwards each message to its destination. The order of forwarding does not depend on the arrival order, *e.g.*, the lexicographic ordering of messages. Figure 1 illustrates the operation of the original mix as described in [Cha81].

As all communication in the system is under surveillance, ambiguity in traffic analysis can only be introduced inside a trusted mix receiving two or more messages from honest vertices. As a result, a mix cannot immediately forward a message it receivers, but rather has to wait until it accumulates a few messages. In particular the protocol has to specify its waiting policy. This point is not addressed in Chaum's original paper. It is also a source for many simple yet powerful attacks on the protocol. Today there are many policies suggested, with attacks, fixes and counter-attacks [Ray00].

Onion Routing - A Cascade of Mixes The second feature of Chaum's anonymity preserving system is the use of public key cryptography in performing routing of messages through a cascade of mixes. The usage of message encryption between any two points in the system inhibits an adversary from following message paths based on their content. This effectively reduces the problem to dealing with traffic analysis.

When using a cascade of mixes the sender randomly chooses a list of mixes through which the message is to be routed. The message is then encrypted multiple times, each time a new encryption "peel" is added. Each additional encryption layer contains the needed information for a specific mix in the cascade. Figure 2 supplies details of cascade mixing and demonstrates how it is used to route messages in a system. This type of message routing is nicknamed "Onion Routing" [GRS96].

The original system devised by Chaum allows returning messages back to the sender without revealing its identity (even to the receiver). This is achieved by having the message include two parts, the regular forward message "onion", and another separate "onion" containing the needed routing information and encryption keys that allows backward return of answers.

What Is the Adversarial Model? Chaum's description of the protocol does not supply a proof of correctness. A natural question is against what type of adversarial model the protocol is secure. Many attempts were made to analyze the strength of the "onion routing" process. Much subsequent work has shown the system to be insecure and highly sensitive to strong adversaries, and in particular the system is problematic when considering adaptive adversaries that control all links in the system [Ray00].

We hope that the ideas and methods presented in this paper will form a strong basis for future rigorous analysis of privacy preserving systems, putting an end to the hack and attack cycle. Following are a few examples of attacks described in the literature.

Mix flood attacks take advantage of the fact that Chaumian-mixes store a number of messages before forwarding each one to its destination. The adversary floods a mix with her messages, making sure they compose the large majority of messages it stores and forces a mix to output its stored messages.

A *replay* attack takes place when a controlled node in the system, or an entity controlling a communication link re-sends one (or more) of the messages that were previously sent through this node/link. This effectively replays a part or all of the followed message's path. The adversary can concentrate attention on one specific path and can use such replays to reproduce

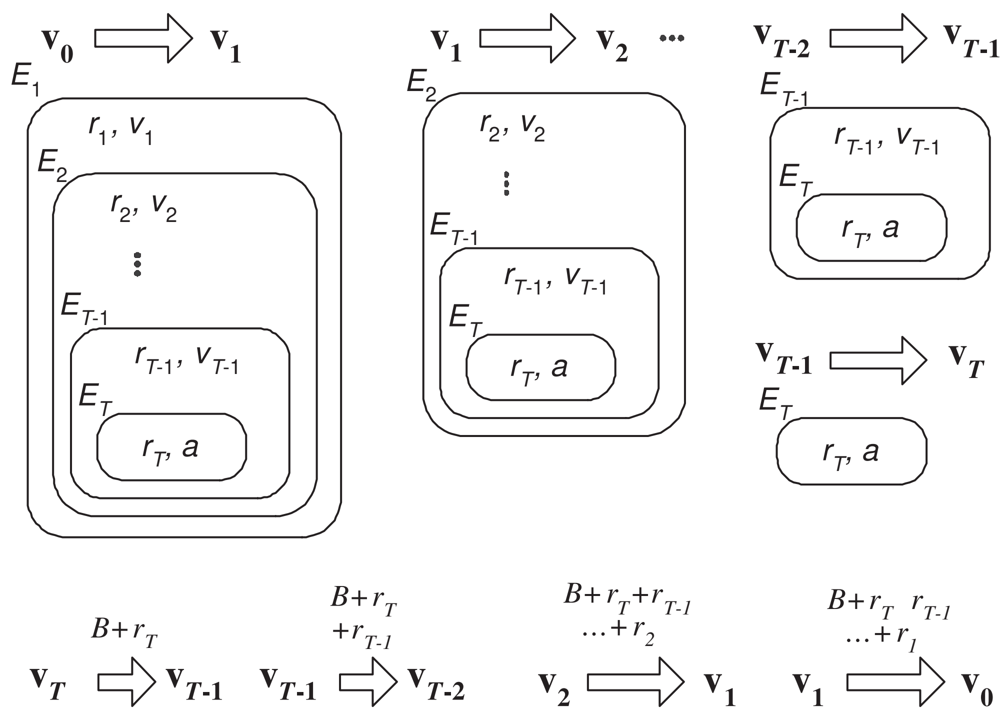

Figure 2: *The cascade of encryptions when sending a message a from v_0 to v_T via the path $v_0 \mapsto v_1 \mapsto v_2 \mapsto \cdots \mapsto v_{T-1} \mapsto v_T$, where E_i is the public key encryption associated with node v_i. Note that this process already prepares the symmetric keys (the r_i strings) used when sending the response b from v_T to v_0, if required, along the reverse path $v_T \mapsto v_{T-1} \mapsto \cdots \mapsto v_2 \mapsto v_1 \mapsto v_0$. The data transmitted along the reverse path is also illustrated above.*

the communication pattern over and over again, making it statistically more prominent than others, thus isolating it. It is easy to find a remedy for this specific type of attack (using time-stamps or message identifiers for example), yet this brings us back to the case of non-general solutions to specific problems.

1.4.3 Rackoff and Simon

Rackoff and Simon propose two algorithms.

The first one is a variant of Chaum's algorithm. As with Chaum's protocol they use node-mixing, and in their attack model is an adaptive adversary that controls *all* links in the system, and any constant fraction of the links. They assume all players play all the time and use secure computation. Under these conditions they show mixing occurs with poly-logarithmic steps (this was later improved to logarithmic number of steps, [CKKL99, CK00]).

Rackoff and Simon [RS93] also describe a simple system based on sorting networks, that is unlinkable against a *passive* adversary that controls all the communication links but *none* of the nodes. They generalize the system to *passive* adversaries that may control some of the nodes using secure multiparty communication. Their system requires all players to participate in each round. The idea is that if the number of honest players is a fixed percentage of all players and if all players play in all rounds of the protocol, then there is a constant probability of having two honest messages arrive at a mix.

The authors extend both solutions to malicious adversaries as well, assuming a public blackboard accessible by all players, and a stopping mechanism for the entire system if malicious behavior is discovered.

The disadvantages of both systems include the requirement that all players participate during each round of the protocol, and the use of secure computation during the protocol.

1.4.4 A Mix-based System Based Over Permutation Networks

Abe ([Abe99]) proposed a mix-based protocol that uses a bulletin board and zero knowledge proofs. The interesting point about Abe's protocol is that it uses networks that have the property that for every given permutation the network's switches can be set so that the network implements the given permutation. In his paper Abe implicitly claims that if one takes a permutation network and uses each switch as a crossover (*i.e.*, each switch with probability half implements identity, and with probability half switches its inputs) then any permutation network perfectly obscures permutations. The claim is however faulty as was shown in [AH01]. There exists a fix, but only under additional requirements that participating mixes coordinate the creation of permutation networks.

1.4.5 Crowds

A somewhat different system is Crowds [RR98]. Crowds has a set of intermediate nodes, some of them may be corrupt, but they are not used as mixes. Instead every node takes a probabilistic decision whether to send the message to its final destination, or to forward it to another intermediate node. Crowds supplies a formal analysis of the security it provides, but the security is very mild (it is proportional to the path length). The efficiency and security of a crowd grows as the number of its members grows. The system is highly applicable in peer to peer networks.

1.4.6 Anonymous Multiparty Computation

Anonymous Multi Party Computation (AMPC) is introduced in [MP01] as a building block for an anonymous communications subsystem which does not utilize any complex cryptographic primitives. The system is robust against malicious modifications of data through executions of the protocol, and its strength is evaluated under a limited adversary model. The delay and load factors of the system highly depend on the specific assumptions regarding the strength of the adversary. The system is applicable for achieving either sender anonymity or receiver anonymity, yet it is not clear if it achieves unlinkability.

1.4.7 Busses

An interesting new approach, quite different from Chaumian mixes, is taken in [BD03]. The system used is analogous to public transportation, where people are hard to track as they move along within large groups of people (*e.g.* a bus). The authors give rigorous proofs of security. Unfortunately, it also has a rather inefficient time/space tradeoff and thus is not “efficient” in our terminology.

1.4.8 Dining Cryptographers Networks

Chaum’s Dining-Cryptographer networks (DC-Nets) [Cha88] allow both provable sender and receiver anonymity and unlinkability, secure against passive adversaries, as well as some stronger forms of adversaries. The scheme uses shared secret keys, requires secure and reliable broadcast mechanisms and all users need to participate at every message delivery, making it inefficient. The system is information theoretic secure, and is a special case of secure computation.

1.4.9 Some mix based implementations

We finally mention some mix-based implementations. The Anonymizer [Ano] is a single “trusted” mix implementation. Electronic Frontiers’ Anonymous Remailer [EFG] is a full implementation of Chaum's ideas, filling in many details missing in Chaum's paper. Onion Routing [SGR97, RSG98] implements these ideas beneath the application level making the underlying communication network seamlessly anonymous.

The Free Haven Project [DFM00] aims to deploy a system for distributed, anonymous, persistent data storage which is robust against powerful adversaries. The project's site [FH] contains a large repository of anonymity related material.

There is a very significant body of work on mix-based system, and a good survey of this massive volume of work is the collection of papers [Fed00].

1.5 Our protocol

The prime objective of this work is to show that there exists a simple, efficient and practical anonymous communication protocol with provable properties.

Chaum suggested an adversary that controls all links. In this attack model mixing can happen only when an honest vertex receives two or more honest messages, which we call *node mixing*. Under this assumption, any secure protocol must have a constant fraction of honest messages among all messages in the system, otherwise in most mixes a honest message is isolated. This forces all such systems (that are secure against adaptive adversaries) to require that every honest player plays at each round, and as a result we lose efficiency.

We do not use node mixing, instead we use a new paradigm that we call *layer mixing*. Layer mixing occurs when two honest vertices communicate with two other honest vertices, using honest links. Layer mixing can happen even when node mixing does not (*e.g.* see Fig. 3). Notice also that if the adversary controls all links in the system, then layer mixing is impossible. Thus, we weaken the adversary model (from the assumption that it controls all links in the system to the assumption that it controls most but not all the links in the system) but gain in that we no longer require that all honest players play at every round.

1.5.1 Our Attack Model

The results presented in this work focus on networks in which the adversary is adaptive. The case of malicious adversaries is left for future work, and is briefly discussed in Section 9.3.

We assume that the system has a time bound Δ_{bound} on message delivery. Every message sent over a communication link arrives within that time bound. We also assume that all players have clocks, and all the clocks are within $\Delta_{accuracy}$ time of some global time, and that a public key infrastructure (PKI), and a public key directory is widely available, *i.e.*, every node

Protocol	Attack Model: Resources under adversary control	Delay	Load	Remarks?
Mix-nets [Cha79]	1% links, α% nodes	polylog(N)	polylog(N)	Adaptive Insecure Malicious Insecure
Mix-nets [RS93]	1% links, α% nodes	polylog(N)	polylog(N)	Adaptive Secure comp Heavy load
Mix-nets +sorting [RS93]	1% links, 0% nodes	polylog(N)	$\tilde{O}(N)$	Passive Heavy load
Mix-nets +sorting [RS93]	1% links, α% nodes	polylog(N)	$\tilde{O}(N)$	Malicious Secure comp Heavy load
DC-nets [Cha88]	1% nodes 0% links	$O(1)$	$O(N^2)$	Passive Shared randomness Broadcast Heavy load Information theoretic
Crowds [RR98]	α% nodes α% links	Dynamic	Delay	Adaptive Weak security
AMPC [MP01]	α% nodes α% links	$O(N)$	$O(N^2)$	Limited adversary Heavy load Heavy delay
Busses [BD03]	α% nodes 1% links	$O(N)$	$O(N^2)$	Heavy load Heavy delay
This paper	α% nodes α% links	polylog(N)	polylog(N)	Adaptive Efficient

Table 1: Unlinkability protocols for a network of size N. Delay is how long it takes for an anonymous message to arrive after it's been initiated. Load is the number of messages actually sent per anonymous message delivered. α% is any fixed fraction (arbitrarily close to 1). "1% links", means all links, "1% nodes" means arbitrary number of malicious nodes. "0% links", means no malicious links, "0% nodes" means no malicious nodes.

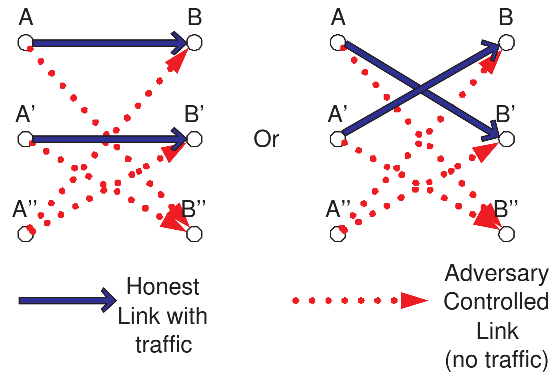

Figure 3: *Layer mixing: ambiguity is introduced because none of the four nodes A, A', B, B' and none of the four links $A \mapsto B$, $A \mapsto B'$, $A' \mapsto B$ and $A' \mapsto B'$ are under adversary control. The adversary is unsure whether A communicated with B or with B'. Notice that although the adversary has not seen any communication in this stage, the adversary knows that A and B together communicated with A' and B', because of links he controls that were not used for communication.*

has a public key. The most significant assumption we make is that at least a constant fraction of the communication links are honest.

1.5.2 The Protocol

Our protocol is very similar to Chaum's [Cha81]. We first describe it in a synchronous system, and in Section 8.1 we explain how to adapt it to an asynchronous system. A wants to send a message $a \in \{0,1\}^S$ to B and get back an answer $b \in \{0,1\}^S$, where S is the length of data items in the system. It is crucial that all data items in the process are of the same length to disallow traffic analysis based on different lengths of items. A picks $T-1$ random nodes $v_1, \ldots, v_{T-1}$, and sets $v_0 = A$, $v_T = B$. A also picks T random strings $r_i \in \{0,1\}^S$, and $z_i \in \{0,1\}^{\ell_i}$ where ℓ_i is a security parameter for the encryption schemes E_i. We let $E_1, \ldots, E_T$ be the public encryption methods of the T nodes. We denote

$$a_i \quad = \quad E_{i+1}(r_{i+1}, z_{i+1}, v_{i+2}, E_{i+2}(\ldots E_{T-1}(r_{T-1}, z_{T-1}, v_T, E_T(r_T, a)))\ldots)$$

for $i = 0, \ldots, T-1$. The cascade of encryptions is depicted in Figure 2, and for clarity we omit the z_i parts.

The way from A to B : A sends $(0, v_0, z_0, a_0)$ to v_1. In general, v_i sends the message (i, v_i, z_i, a_i) to v_{i+1}. When v_{i+1} receives (i, v_i, z_i, a_i) where

$$a_i = E_{i+1}(r_{i+1}, z_{i+1}, v_{i+2}, a_{i+1})$$

it decrypts a_{i+1}, and sends $(i+1, v_{i+1}, z_{i+1}, a_{i+1})$ to v_{i+2}. It also records $v_i, v_{i+2}, z_i, z_{i+1}$ and r_{i+1}. $v_T = B$ recognizes it is the last on the path, and prepares an answer $b \in \{0,1\}^S$ to the message a it receives.

The way back : $B = v_T$ sends (v_T, z_{T-1}, b_T) to v_{T-1} where $b_T = b \oplus r_T$. In general, v_i receives a message (v_{i+1}, z_i, b_{i+1}). v_i recognizes the value z_i, the link (v_{i-1}, v_i) that precedes (v_i, v_{i+1}) and the values r_i, z_{i-1} that are associated with it. It then sends $(v_i, z_{i-1}, b_i = b_{i+1} \oplus r_i)$ to v_{i-1}. Finally, $A = v_0$ receives $(v_0, z_0, b_0 = b_1 \oplus r_1)$ from v_1. The value $b_0 \oplus r_1 \oplus \dots r_T$ is the desired value b.

Remarks:

1. The role of the z_i fields is as follows. First, it makes the encryption scheme probabilistic and second, if the paths of messages coincide, the z_i's are used to distinguish them.

2. The keys r_i are used on the way back as one time pads for the transferred data, and therefore need to be as long as data items. It is, however, possible to use any other symmetric encryption algorithm with constant short-length keys, *e.g.*, DES or AES, assuming it is secure against the adversary. When measuring the data overhead imposed by our protocol on the length of a single message we notice that using short-length keys as suggested, only a constant amount of data is added to each data item at each round, regardless of the data item size in the system. For paths of T rounds and data items of S bits, the size of messages is therefore $O(S+T)$ bits rather than $O(S \cdot T)$.

3. Adaptive adversaries cannot change the path of return messages, and any vertex that forwards a message, knows in advance when that message will come back to him in the reverse direction. This implies that for traffic analysis we can ignore the return paths and analyze only the forward paths.

We formally define unlinkability in section 2

Theorem 1.1. *Assume the above protocol runs in a network with N nodes, $\binom{N}{2}$ communication links, some constant fraction of which are honest, then the protocol is $\alpha(N)$–unlinkable when $T \geq \Omega(\log(N)\log^2(N/\alpha(N)))$.*

We picture a possible execution of the protocol in Figure 4.

1.5.3 Paper overview

We give information theory background in Appendix A. We use standard notations and definitions, as appearing in [CT91] and in [NC00]. In Section

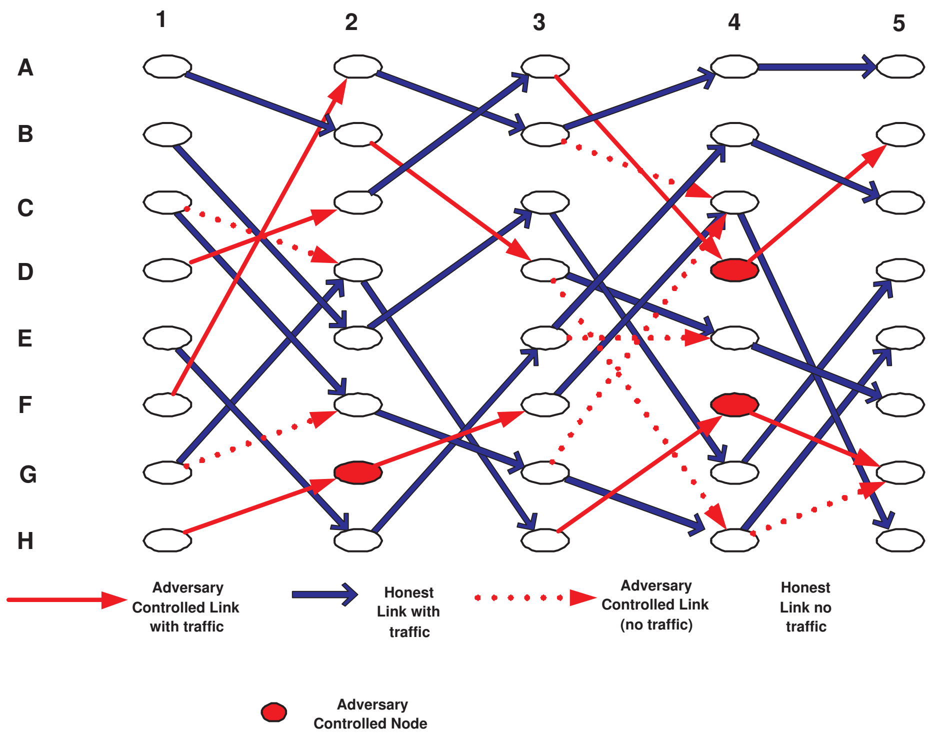

Figure 4: *Adversary controlled nodes imply that the adjacent links are also adversary controlled. We drew only the actual communication links used (with solid lines) going in and out of adversary controlled nodes. The nodes at coordinates* (x, y) *and* (x', y') *are not necessarily distinct.*

2 We present the former formal definition of unlinkability and give a new definition that uses the information theoretic mutual information function. We show the new definition is equivalent to the old one.

Protocol executions are usually pretty messy, as can be seen in Fig. 4. Our goal is to find a simple structure (which we call an "Obscurant Network", see Section 3) inside the global mess and prove that this structure by itself suffices for obscuring the messages paths. In Section 3 we discuss obscurant networks. In section 4 we define a safe embedding of a general (not necessarily obscurant) network into an execution of a protocol. The embedding definition has several subtleties. In section 5.2 we show that one can always safely embed a *shallow* network, into a protocol that takes more (but not much more) time than the depth of the network. Finally, in Section 6 we prove this implies our protocol is unlinkable, in the no-prior information case.

Prior information is dealt with in Section 7. A folding trick is used to reveal yet some more information to the adversary about the connections between information flowing from the sources of the message and the information arriving at the final destinations of messages. This trick literally folds the communication pattern in half when observed from the adversary's point of view, reducing the analysis to the case of no-prior information, when the interesting layer of the protocol is the middle layer of communications. We then show that the middle layer does not convey enough information to the adversary, resulting in unlinkability.

Section 8 extends our results to the asynchronous setting as well as to somewhat stronger adversaries, and in section 9 we summarize our work and discuss open problems and future work.

2 Defining Unlinkability

We begin the with unlinkability definition of Rackoff and Simon. Their definition captures also the prior information case (even though their protocol applies only to the no-prior information case).

Say there are M active players that wish to communicate with M distinct nodes[2]. Let Π be the random variable that describes the *a-priori* distribution over communication patterns. I.e., in an actual protocol, π will be chosen according to the distribution Π, and then player i will use the protocol to send a message to player $\pi(i)$. If we have no prior information on the players' communication patterns, Π is the uniform distribution over all $M!$ possible permutations. Now, let C be correlated the random variable whose value is all the information available to the adaptive adversary, gathered from adaptive communication links and adaptive nodes. Specif-

[2] If the M nodes are not distinct, then our protocol w.h.p. makes them distinct by adding a random identifier to each message. Protocol details are found in Section 1.5.2.

ically, C is a 0/1 matrix with rows indexed by time steps t and columns indexed by communication links e, and with $C_{t,e}$ being 1 iff there is some communication on edge e in time t. Simon and Rackoff require that (Π, C) is α–computationally close to some (Π, C') such that for all $\pi_1, \pi_2 \in S_M$ $|(C'|\Pi = \pi_1) - (C'|\Pi = \pi_2)|_1 \leq \alpha$.

In Section 2.1 we first generalize this definition to unlinkability between two random variables, and give this definition both in the information theoretic world and the computational world (Section 2.2). We also give a definition using the mutual information function, and claim it is equivalent to the first definition (the proof appears in Section 2.5). In Section 2.3 we specialize to communication protocols, and define efficiency (Section 2.4).

2.1 Unlinkability of two random variables

Let A, B be a family of two correlated random variables $\{(A, B)\} = \bigcup_n (A_n, B_n)$. We now suggest three alternative definitions to unlinkability.

Definition 2.1. (information theoretic unlinkablity) A family of distributions $\{(A, B)\} = \bigcup_n (A_n, B_n)$ is information theoretically $\alpha = \alpha(n)$–unlinkable if for every n,

- (Def 1): $\Pr_{a \in A_n} \; [\; |(B_n|A_n = a) - B_n|_1 \geq \alpha \;] \;\leq\; \alpha.$
- (Def 2): $\Pr_{b \in B_n} \; [\; |(A_n|B_n = b) - A_n|_1 \geq \alpha \;] \;\leq\; \alpha.$
- (Def 3): $I(A_n : B_n) \;\leq\; \alpha.$

We prove that all three definitions are essentially equivalent. Formally,

Lemma 2.1. *(equivalence of definitions) let $\{(A, B)\} = \bigcup_n (A_n, B_n)$ be a family of arbitrary joint distributions, (A_n, B_n) is distributed over some domain Λ_n.*

- *If $\{(A, B)\}$ is information theoretically $\gamma(n)$–unlinkable according to Definition 1 or Definition 2, then it is information theoretically $\delta(n) = O(\log(|\Lambda_n|)\sqrt{\gamma(n)})$–unlinkable according to Definition 3.*
- *Conversely, If $\{(A, B)\}$ is information theoretically $\delta(n)$–unlinkable according to Definition 3, then it is information theoretically $\gamma(n) = (2 \ln 2 \cdot \delta(n))^{1/3}$–unlinkable according to Definition 1 or Definition 2.*

2.2 The Computational Setting

Definition 2.1 is in an information-theoretic setting. In our case of anonymous communication, we reduce the linkability question to traffic analysis by claiming that the encryption scheme we use is computationally secure.

This forces us to work with a definition applicable in the computational setting. The solution for that is standard (and was also used by [RS93]). We first define computational indistinguishability:

Definition 2.2. (Computational indistinguishability) Let $A = \{A_n\}, B = \{B_n\}$ be two families of distributions. We say $d(A,B)_P \leq \delta(n)$, if for every family of polynomial-size Boolean circuits $\{T_n\}$, for every large enough n,

$$| \Pr_{x \in A_n}[T_n(x) = 1] - \Pr_{x' \in B_n}[T_n(x') = 1] | \quad \leq \quad \delta(n)$$

Having defined computational indistinguishability, we can now define computational unlinkability:

Definition 2.3. (computational unlinkability) A family of distributions $\{(A,B)\} = \bigcup_n (A_n, B_n)$ is $\alpha(n)$*–unlinkable* if, (A,B) is $\alpha(n)$-computationally close to some (A', B') and (A', B') is information theoretically $\alpha(n)$-unlinkable.

With these definitions, one can phrase a computational analogue to Lemma 2.1, this computational variant also holds.

2.3 Unlinkable Communication Protocols

We now specialize to communication protocols.

We now define unlinkability for communication protocols. Our definition is general and covers scenarios where the adversary has prior information.

Definition 2.4. ($\alpha(N)$–unlinkability) We say a protocol $P = \{P_N\}$ is $\alpha(N)$–unlinkable according to definition i, $i \in \{1,2,3\}$, if for every N players, every choice of a subset H of honest players, and every distribution $\Pi_N = \Pi_N(H)$ on their actual communication (Π is the prior knowledge) if we let $C_N = C(H)$ be the correlated random variable that contains the information known to the adversary, then $\bigcup_N (\Pi_N, C_N)$ is $\alpha(N)$– unlinkable according to definition i.

We mention that previous definitions did not allow prior information. This omission is explicit in the work of Rackoff and Simon, and implicit in the vast body of work on "applied" protocols. It seems clear that the assumption the adversary has no prior information is typically false, *e.g.*, an adversary might know that residents of China tend to correspond with other residents of China. We also note that the prior knowledge the adversary has can have — in fact is likely to have — very small weight in a uniform world. *E.g.*, in a uniform world it is very unlikely that residents of Beijing correspond much with each other. We discuss prior knowledge in depth in Section 7. We believe that any reasonable definition that portends to say something about the real world should deal with prior knowledge.

We have

Theorem 2.2. *If a protocol P is $\alpha(N)$–unlinkable according to any one definition then it is $\alpha'(N) = N^3\alpha(N)^{1/3}$–unlinkable according to any other definition.*

Proof. Let $P = \bigcup_N P_N$ be an $\alpha(N)$–unlinkable protocol according to some definition i. Let $|\Lambda_N|$ be the size of the domain on which (Pi_N, C_N) is distributed. We soon show that $\log(|\Lambda_N|) \leq O(N^3)$. By Lemma 2.1, P is $\alpha'(N)$–unlinkable according to definition j.

We now bound $\log(|\Lambda_N|)$. The random variable C_N tells for each dishonest link in the protocol whether there was communication on that link or not. There are $M \leq N$ vertices and so at most N^2 links per layer, and DN^2 links overall in a protocol of length D. For every such link, C_N may contain a zero/one bit whether there was traffic on that link or not. Altogether C_N has support size 2^{DN^2}. Π_N has a smaller support. This implies that $\log(|\Lambda_N|) = O(DN^2)$. However, as noted above $D \leq T \leq N$. To summarize, $\log(|\Lambda_N|) \leq O(N^3)$. □

2.4 Efficient Protocols

We now define efficiency:

Definition 2.5. (efficiency) A protocol P on N players, designated to give $\alpha(N)$ unlinkablity, is *efficient* if the protocol takes $T(N) = O(\text{poly}(\log(\frac{N}{\alpha(N)})))$ rounds, and $O(M \cdot T(N))$ messages, where M is the number of honest players who wish to send a message at a time.

2.5 Equivalence of Definitions

Before proving the equivalence we prove a lemma.

Lemma 2.3. *Let A and B be two arbitrary random variables,* $0 \leq \alpha \leq 1$. *Then*

$$\Pr_{b \in B}[\ |A - (A|B=b)|_1 \geq \alpha] \quad \leq \quad 2\ln 2 \cdot \frac{I(A:B)}{\alpha^2}$$

Proof. The proof uses the relative entropy function. For details see Section A.2.

$$\begin{aligned}
\Pr_{b \in B}[\ |A - (A|B=b)|_1 \geq \alpha] \quad &= \quad \Pr_{b \in B}[\ \frac{1}{2\ln 2} \cdot |A - (A|B=b)|_1^2 \ \geq \ \frac{1}{2\ln 2}\alpha^2] \\
&\leq \quad \Pr_{b \in B}[\ D((A|B=b)\ ||\ A) \geq \frac{1}{2\ln 2}\alpha^2] \\
&\leq \quad \frac{\mathbb{E}_{b \in B} D((A|B=b)\ ||\ A)}{\frac{1}{2\ln 2}\alpha^2}
\end{aligned}$$

where the first inequality is by Fact A.6 and the second one is Markov inequality. Thus, it suffices to prove that $\mathbb{E}_{b\in B}\ D((A|B=b)\ ||\ A) = I(A:B)$. Indeed,

$$\begin{aligned}
& \mathbb{E}_{b\in B}\ D((A|B=b)||A) \\
= & \sum_b \Pr(B=b) \sum_a \Pr(A=a|B=b)\cdot \log\left(\frac{\Pr(A=a|B=b)}{\Pr(A=a)}\right) \\
= & \sum_{a,b} \Pr(B=b)\Pr(A=a|B=b)\cdot \log\left(\frac{\Pr(A=a|B=b)\Pr(B=b)}{\Pr(A=a)\Pr(B=b)}\right) \\
= & D((A,B)||A\otimes B) = I(A:B)
\end{aligned}$$

□

We now prove Lemma 2.1:

Proof. Fix n and set $A = A_n$, $B = B_n$.

Definition 3 → Definition 2: Suppose $I(A:B) \le \delta$. Set $\gamma = (2\ln 2\cdot\delta)^{1/3}$. We use Lemma 2.3 to get

$$\Pr_{b\in B}[\ |A-(A|B=b)|_1 \ge \gamma\ ] \le 2\ln 2\cdot\frac{I(A:B)}{\gamma^2} \le 2\ln 2\cdot\frac{\delta}{\gamma^2} = \gamma.$$

Definition 3 → Definition 1 follows the same arguments simply by noting that $I(A,B) = I(B,A)$.

Definition 2 → Definition 3: We assume $\Pr_{b\in B}[\ |(A|B=b)-A|_1 \ge \gamma\ ] \le \gamma$. We compute the statistical distance between (A,B) and $A\otimes B$. We see that:

$$\begin{aligned}
d &= |(A,B) - A\otimes B|_1 \\
&= \sum_b\sum_a |Pr(A=a\ |\ B=b\ )\cdot Pr(B=b) - Pr(A=a)\cdot Pr(B=b)| \\
&= \sum_b Pr(B=b)\cdot|(A|B=b)-A|_1\ \le\ 1\cdot\gamma + 2\cdot\gamma \le 3\gamma.
\end{aligned}$$

We now use Fact A.4 with (A,B) and (A',B') chosen from $A\otimes B$. Because $H(A\otimes B) = H(A)+H(B)$ it follows that $I(A',B')=0$ so we get that

$$|I(A:B) - I(A',B')| = I(A:B) \quad\le\quad 3\cdot(3\gamma\cdot\log(|\Lambda|) + 3\gamma\cdot\log(\frac{1}{3\gamma}))$$

It follows that $I(A:B) \le \delta = O(\log(|\Lambda|)\sqrt{\gamma})$.

Definition 1 → Definition 3 is similar.

□

3 Obscurant Networks

A network is a layered circuit with the same number of vertices on each layer. We say a circuit is a crossover network, if every vertex has in-degree and out-degree one or two, and the circuit consists of crossover gates and simple connections only. An example of such a network is depicted in Figure 6.

We think of the following game: a pebble is put on some input vertex, say on the i'th vertex. If the vertex out-degree is one, we follow that link. Otherwise, we follow each of the crossover links with probability half. By the end of the game we get a distribution O_i over the output elements. We say the network ϵ–obscures the i'th input, if $|O_i - U_M| \leq \epsilon$. We say a network ϵ-obscures *inputs*, if it ϵ-obscures every input.

We can generalize this notion to crossover networks that obscure *permutations*. We think of the game where there are M pebbles, numbered from 1 to M. If a pebble lies on a vertex whose out-degree is one, the pebble follows that link. Otherwise, if two pebbles share a crossover, we choose each of the two possible matchings with equal probability and follow the matching. By the end of the game we get a distribution O over all possible permutations. We say the network ϵ–obscures permutations, if $|O - U_{M!}| \leq \epsilon$.

It is not true that a network that obscures inputs must also obscure permutations. An example for that is the butterfly network B_n which perfectly obscures inputs but does not obscure permutations. Notice that if we know where pebble 1 goes to, we also know that pebble 2 can go only to a subset of size $n/2$ of possible destinations. However, non-explicit shallow networks that obscure permutations exist:

Theorem 3.1. *(Non-explicit permutation networks) [CKKL99, CK00] For every n, there exists non-explicit networks over n inputs that ϵ–obscure permutations, and whose depth is $O(\log(\frac{n}{\epsilon}))$.*

There are also several explicit constructions with poly $\log(n)$ depth [Mor05], but non-explicit constructions are as good for us as explicit constructions.

3.1 Constructing Shallow Networks That Obscure Inputs

In this section we call a network obscurant if it obscures *inputs*. We now build a shallow obscurant network on M inputs, for any given fixed integer M.

Let Z be the largest power of two not larger than M, *i.e.* $\frac{M}{2} < Z \leq M$. In the construction we use two components: a butterfly network B_Z with Z vertices in each layer, with comparators replaced with crossovers, and a

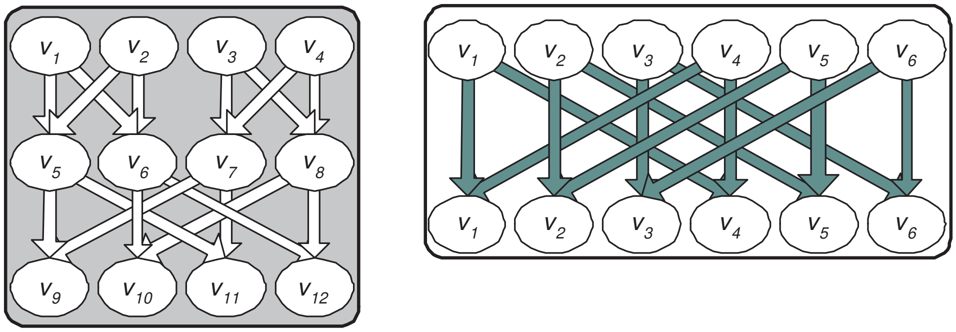

Figure 5: Example of components used to build an obscurant network. The left network is a butterfly with $Z = 4$ elements, B_4. The right network is P_{2k} with $k = 3$.

network with two layers and $2k$ elements in each layer, with k crossovers connecting vertices i and $k + 1 \pmod{2k}$ in the first layer with vertices i and $k + i \pmod{2k}$ in the second layer, for $i = 1, \ldots, 2k$. We call this later network P_{2k}. Figure 5 illustrates an example of these components.

We distinguish between two cases. If $Z = M$ we put B_Z on the Z inputs. Otherwise, $\frac{M}{2} < Z \leq M$. We then do the following:

- For the first level, we put B_Z on the Z rightmost elements.
- For the second level, We put B_Z on the Z leftmost elements.
- For the third level, We put $P_{2(M-Z)}$ on the $2(M - Z)$ rightmost elements.
- For the fourth level, we put B_Z on the Z leftmost elements.
- We then iterate the third and fourth levels $\log(M) + \log \epsilon^{-1}$ times. See Figure 6.

We claim:

Lemma 3.2. *If the depth D of the network is $O((\log(M) + \log \epsilon^{-1}) \log(M))$, the network is ϵ–obscurant.*

Proof. If $M = Z$, then for every input vertex, i spans a tree. It follows that $O_i = U_M$ and the network is 0–obscurant.

Suppose $\frac{M}{2} < Z < M$. Let i be a starting vertex. Notice that B_Z gives equal weight to each of its Z outputs. It therefore follows that after the application of B_Z on the right followed by B_Z on the left, all the leftmost Z elements have one weight, ℓ_0, and all the remaining $M - Z$ (rightmost)

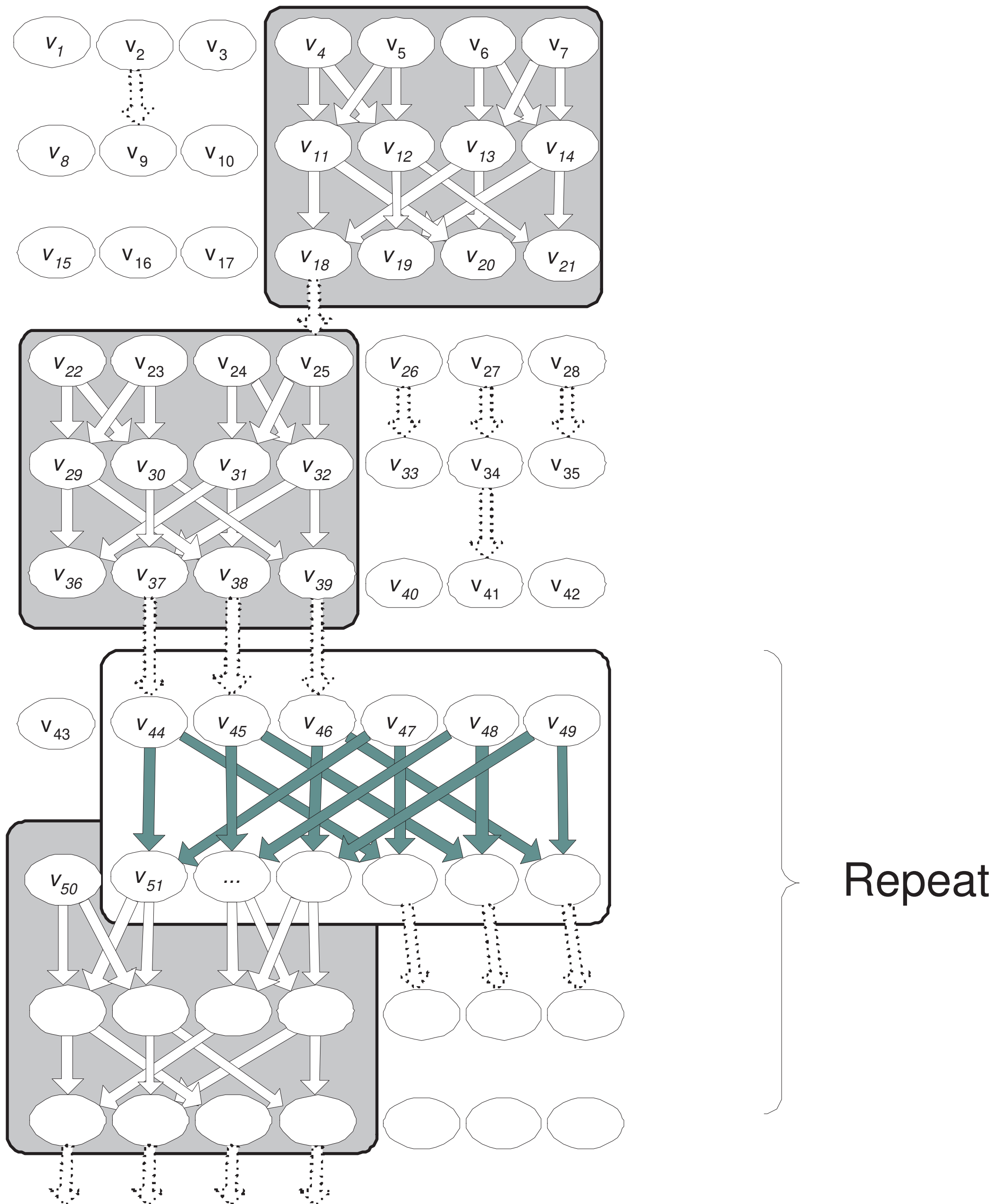

Figure 6: *An obscurant network for* $M = 7$, $Z = 4$. *Simple connections are often omitted. The boxes with the grey background are* B_4 *butterflies. The other boxed sub-circuit is* P_6.

elements have the same (possibly different) weight, r_0. One can observe that this property is an invariant that remains valid throughout the protocol, *i.e.*,

- The invariant: After applying the pair of transformations $P_{2(M-Z)}$ and B_Z i times, $i \geq 0$, the Z leftmost elements all have weight l_i and the $M-Z$ rightmost elements all have weight r_i.
- After applying the $P_{2(M-Z)}$ network of the $i+1$'st pair, all the $2(M-Z)$ rightmost elements have one weight $r_{i+1} = (r_i + l_i)/2$ and all the remaining $M - 2(M-Z) = 2Z - M = K$, $1 \leq K < Z$, leftmost elements remain at weight l_i.
- After applying the B_Z network of the $i+1$'st pair, all the Z leftmost elements have one weight $\ell_{i+1} = \frac{K \cdot l_i + (Z-K) \cdot r_{i+1}}{Z}$ and all the $M-Z$ rightmost elements remain at weight r_{i+1}.

Our goal is to show that ℓ_i approaches r_i quickly. Indeed,

$$
\begin{aligned}
|\ell_{i+1} - r_{i+1}| &= \left| \frac{K \cdot \ell_i + (Z-K) \cdot r_{i+1}}{Z} - \frac{r_i + \ell_i}{2} \right| \\
&= \left| \frac{(Z-K)r_i + (Z+K)\ell_i}{2Z} - \frac{Zr_i + Z\ell_i}{2Z} \right| \\
&= \left| \frac{K}{2Z}(\ell_i - r_i) \right| \; < \; \frac{1}{2}|\ell_i - r_i|.
\end{aligned}
$$

The last inequality follows as $K < Z$.

As $|\ell_0 - r_0| \leq 1$ it must be that $|\ell_t - r_t| < 2^{-t}$. We see that $1 = Z\ell_t + (M-Z)r_t < Z\ell_t + (M-Z)(\ell_t + 2^{-t}) = M\ell_t + (M-Z)2^{-t}$ and so $\ell_t \geq \frac{1-(M-Z)2^{-t}}{M}$. We similarly show that $\ell_t \leq \frac{1+(M-Z)2^{-t}}{M}$. Together, $|\ell_t - \frac{1}{M}| \leq \frac{(M-Z)}{M}2^{-t}$. To conclude the proof we note that $|O_i^{(t)} - U_M|_1 \leq 2M \cdot |\ell_t - \frac{1}{M}| \leq 2(M-Z)2^{-t}$. As $M - Z < M/2 < M$, we get that $|O_i^{(t)} - U_M|_1 \leq M2^{-t} \leq \epsilon$, for $t = \log(M) + \log \epsilon^{-1}$. □

4 Embeddings and Safe Embeddings

4.1 Embedding a Network Into an Execution of a Protocol

An embedding of a graph $G_1 = (V_1, E_1)$ into another graph $G_2 = (V_2, E_2)$ is a mapping $\phi : V_1 \to V_2$ such that for every $(a, b) \in V_1$ we have $(\phi(a), \phi(b)) \in V_2$ (this notion is called a graph homomorphism).

In our case the two graphs are: G the given network, and P an actual execution of the protocol. The bottom level of P is the initial stage of the protocol, and choosing M honest players corresponds to an embedding of the M vertices in the bottom level of G to M distinct honest vertices in the (much larger) bottom level of P. This choice is adversarial (because the protocol should protect any subset of M honest players) and so is given to us as input.

One property that we would like to maintain in our embedding, is that the embedding is consistent from the adversary point of view. That is, if (a,b) is an edge in G and $v = \phi(a)$, $w = \phi(b)$, then from the adversary point of view it is possible that indeed v was routed to w.

The consistency property in particular implies that if $\{a,b\}$ is connected via a crossover to $\{c,d\}$ in G, and $v_1 = \phi(a), v_2 = \phi(b), v_3 = \phi(c)$ and $v_4 = \phi(d)$, then from the adversary point of view both the routing ($v_1 \to v_3$, $v_2 \to v_4$) and the routing ($v_1 \to v_4$, $v_2 \to v_3$) are possible. That is, all the four edges $\{(v_1,v_3),(v_1,v_4),(v_2,v_3),(v_2,v_4)\}$ are not under adversarial control. Intuitively, this is where mixing happens and we gain anonymity.

We embed a depth M network into a depth T graph, where T is much larger than M, and the reason is that we may take many levels to implement an embedding of a single level of G. For example, say $\{a,b\}$ is connected with a crossover in G to $\{b,d\}$ and $\phi(a) = v_1$, $\phi(b) = v_2$. If there are vertices v_3, v_4 in P such that the vertices $\{v_1,v_2\}$ are routed as a set to $\{v_3,v_4\}$, and all four edges between $\{v_1,v_2\}$ and $\{v_3,v_4\}$ are not under adversarial control, then we can map c to v_3 and d to v_4 (and we say $\{v_1,v_2,v_3,v_4\}$ form a crossover in P). If however, no such vertices $\{v_3,v_4\}$ exist, we choose to wait, i.e., we follow v_1 to the vertex v_1' it was routed to in P, and similarly follow v_2 to the vertex v_2' it was routed to in P, and we see if there is a crossover in P over v_1', v_2'. When we eventually find the embedding $v_3 = \phi(c)$ and $v_4 = \phi(d)$, we have the property that there is a crossover between $\{v_1', v_2'\}$ and $\{v_3,v_4\}$ and that v_1 was routed to v_1' and v_2 to v_2' in P. We now give formal definitions.

The setting is that we are given:

- A network G over M inputs and of depth D. We represent G as $G = (V_G, \boxtimes_G, I_G)$ where V_G is the set of DM vertices of G, $\boxtimes_G$ is the set of all groups of four vertices $(a,b;c,d)$ such that (a,c), (a,d), (b,c), (b,d) form a crossover in G, and I_G is the set of all edges of G not participating in any crossover.

- An execution P of a T step communication protocol involving M honest players. We represent P as $P = (V_P, T_P, C_P)$ where V_P is the set of TM vertices participating in the protocol, T_P is the set of all links carrying traffic in the execution of the protocol, and C_P the set of all links that are under adversary control (whether used to carry traffic or not).

The embedding ϕ maps each vertex $v \in V_G$ to a starting node $\phi(v,0)$ and ending node $\phi(v,1)$. Intuitively, an edge (a,b) in G is mapped to the edge $(\phi(a,1), \phi(b,0))$ in P, and for every vertex $v = \phi(a,0)$ in P, the vertex $v = \phi(a,0)$ was routed to $\phi(a,1)$. Formally,

Definition 4.1. (Embedding) A function $\phi : V_G \times \{0,1\} \to V_P$ is an embedding if:

1. (Starting nodes are routed to ending nodes in P). For every $a \in V_G$, $\phi(a,0)$ and $\phi(a,1)$ are connected in T_P.

2. (Direct edges respect routing in P). Each direct (non crossover) link in G is mapped to a direct connection in P. I.e., for every $e = (a,b) \in I_G$

$$(\phi(a,1), \phi(b,0)) \quad \in \quad T_P.$$

3. (Crossovers). Each crossover in G is mapped to a honest crossover in P. I.e., for every $(a,b;c,d) \in \boxtimes_G$

$$\begin{aligned} \left| \left\{ \begin{array}{ll} (\phi(a,1), \phi(c,0)), & (\phi(a,1), \phi(d,0)), \\ (\phi(b,1), \phi(c,0)), & (\phi(b,1), \phi(d,0)) \end{array} \right\} \cap T_P \right| &= 2, \text{ and,} \\ \left| \left\{ \begin{array}{ll} (\phi(a,1), \phi(c,0)), & (\phi(a,1), \phi(d,0)), \\ (\phi(b,1), \phi(c,0)), & (\phi(b,1), \phi(d,0)) \end{array} \right\} \cap C_P \right| &= 0 \end{aligned}$$

Condition 1 guarantees that starting nodes $\phi(v,0)$ are connected to ending nodes $\phi(v,1)$ in P. Condition 2 guarantees that direct links are reflected in our embedding into P. Condition 3 guarantees that crossovers are reflected in our embedding, and moreover we require that the adversary *does not know* any link in any crossover. This is where we are going to gain anonymity.

Definition 4.2. (Embedding strategy) Let G be a crossover network of depth D and width M. Let $\mathcal{P}$ be a protocol over M players and T steps. An *embedding strategy* for the protocol, with ϵ error, is an algorithm that given an execution $P = (V_P, T_P, C_P)$ of the protocol, outputs a function $\phi_P : V_G \times \{0,1\} \to V_P$ such that:

$$\Pr_{\text{coins of } P}[\phi_P \text{ is an embedding}] \geq 1 - \epsilon.$$

4.2 Safe Embeddings

By definition, any embedding between a graph G and an execution of a protocol P depends on C_P and T_P and hence on the execution of P. However, to be useful we want that given an embedding, and an adversary that knows all traffic on all edges except on crossover edges, many different executions of the protocol are possible. We say an embedding ϕ is *safe* if ϕ's action on the crossovers is independent of the communication that took place on the embedded copy of G. We now give a formal definition.

We define $\boxtimes_P = \phi_P(\boxtimes_G)$ to be the image of $\boxtimes_G$ under the embedding ϕ in the execution P. *I.e.*, the set of all $(v_1, v_2; v_3, v_4) \in V_P^4$ for which there exist $(a_1, a_2; a_3, a_4) \in \boxtimes_G$ s.t. $\phi(a_1, 1) = v_1$, $\phi(a_2, 1) = v_2$, $\phi(a_3, 0) = v_3$ and $\phi(a_4, 0) = v_4$.

Definition 4.3. (Safe embedding) An embedding ϕ from $G = (V_G, \boxtimes_G, I_G)$ to $P = (V_P, T_P, C_P)$ is a *safe* embedding if ϕ is an embedding and for every two executions P and P' that use the same sets of vertices, if $T_{P'}$ agrees with T_P on all edges not participating in $\boxtimes_P$ then $\phi_P = \phi_{P'}$.

Having that we define a *safe* embedding strategy by:

Definition 4.4. Safe embedding strategy) Let G be a crossover network of depth D and width M. Let $\mathcal{P}$ be a protocol over M players and T steps. A *safe* embedding strategy for the protocol, with ϵ error, is an algorithm that given an execution $P = (V_P, T_P, C_P)$ of the protocol, outputs a function $\phi_P : V_G \times \{0, 1\} \to V_P$ such that:

$$\Pr_{\text{coins of } P}[\phi_P \text{ is a safe embedding}] \quad \geq \quad 1 - \epsilon.$$

5 Good, Safe Embedding Strategies Exist

We prove that *any* depth D crossover network with M inputs, has a safe embedding strategy into executions of our protocol, with small error, as long as the number of steps T in the protocol is large enough. Formally,

Theorem 5.1. *Let G be any crossover network with M inputs and depth D. Let us run the protocol $\mathcal{P}$ of Section 1.5.2 with M honest nodes and for $T = 2kD$ steps. Assume the adversary controls at most a $1 - f$ fraction of the edges. Then there exists a safe embedding strategy from G to $\mathcal{P}$ with error at most $\epsilon = DM(1 - f^4)^k$.*

5.1 A Combinatorial Lemma

We begin with a combinatorial lemma. The basic fact that we know about our system is that at least an f fraction of the links are honest. *I.e.*, if we let V be the set of all nodes and E the set of all honest links, then $|E| \geq f \cdot \binom{|V|}{2}$, $0 < f \leq 1$. The following combinatorial lemma asserts that if we choose four vertices v_1, v_2, v_3, v_4 at random from V, then there is a crossover structure on the four vertices with probability at least f^4.

Fact 5.2. *([Alo01], Corollary 2.1) Let $G = (V, E)$ be a graph and assume $|E| \geq f \cdot \binom{|V|}{2}$. Then,*

$$\Pr_{v_1,v_2,v_3,v_4 \in V} [\ \{(v_1, v_3), (v_1, v_4), (v_2, v_3), (v_2, v_4)\} \subseteq E\ ] \quad \geq \quad f^4.$$

5.2 An Embedding Strategy

Let G be a crossover network of width M and depth D. Let $\mathcal{P}$ be the protocol from Section 1.5.2 with $T = 2kD$. We are given an execution $P = (V_P, T_P, C_P)$ of the protocol, and a mapping ϕ from the bottom layer of G to the M honest vertices at the bottom layer of P. We now define an embedding $\phi_P : V_G \times \{0, 1\} \to V_P$ as follows.

5.2.1 Labelling the graph G

We first label the vertices of the graph G in a way that reflects the layer depth and respects the edges of G, i.e., we label a vertex a by $a_i^{(d)}$, where d is the depth of a in G ($0 \leq d \leq D$) and $i \in \{1, \ldots, M\}$, such that:

- If (a, b) is a simple edge in G, and a is labelled $a_i^{(d)}$ then b is labelled $a_i^{(d+1)}$, and,
- If $\{a, b\}$ and $\{c, d\}$ form a crossover in G, and a and b are labelled as a set with $\left\{a_i^{(d)}, a_j^{(d)}\right\}$, then $\{c, d\}$ are labelled as a set with $\left\{a_i^{(d+1)}, a_j^{(d+1)}\right\}$.

An example to such a labelling appears in Figure 7.

5.2.2 Labelling the vertices V_P

We label $\phi(a^{(0)})_i$ by $v_i^{(0)}$. We do not care about the labelling of the other vertices in the bottom layer in P. In general, we are only interested in labelling vertices of P that carry transportation on the M routes chosen by the honest players.

We then go over the layers of G, one by one. For each layer (of the $D+1$ layers of G) we set $2k$ layers in P. The even layers will serve for the mapping itself. The purpose of the odd layers will be explained later.

The odd layers are simple and we first describe them. Say we are now dealing with the d'th layer of G, and with an odd layer t in P. For every edge $(v_i^{(t-1)}, w) \in T_P$ we label w with $v_i^{(t)}$. We do not care about other vertices in that level of P.

Next, we describe the labelling of the even layers. Say we are mapping the d'th layer in G. For every $i \in \{1, \ldots, M\}$ we first set $done(i) = false$. We now look at $a_i^{(d)}$ in G and see if it participates in a direct link of G or a crossover. Then we go over all even layers $t = 2kd + \ell$ for $\ell = 0, \ldots, 2k$ and do the following:

$a_i^{(d)}$ **belongs to a crossover** : If $done(i)$ = false and $a_i^{(d)}$ belongs to a crossover in G, i.e., $(a_i^{(d)}, a_j^{(d)}; a_i^{(d+1)}, a_j^{(d+1)}) \in \boxtimes_G$, then let w and z be the vertices such that $(v_i^{(t-1)}, w), (v_j^{(t-1)}, z) \in T_P$.

- If all these four edges are honest we:
  - Label $\{w, z\}$ with the labels $\left\{v_i^{(t)}, v_j^{(t)}\right\}$ in an *arbitrary* order (say, by the natural order on i and j as numbers),
  - Set $\phi_P(a_i^{(d)}, 1) = v_i^{(t-1)}$, $\phi(a_i^{(d+1)}, 0) = v_i^{(t)}$, $\phi_P(a_j^{(d)}, 1) = v_j^{(t-1)}$and $\phi(a_j^{(d+1)}, 0) = v_j^{(t)}$,
  - Set $done(i) = done(j) =$ true.

  We also say in this case that we have found the crossover $(a_i^{(d)}, a_j^{(d)}; a_i^{(d+1)}, a_j^{(d+1)}) \in \boxtimes_G$ in P.
- Otherwise, one of the edges $(v_i^{(t-1)}, w)$, $(v_i^{(t-1)}, z)$, $(v_j^{(t-1)}, w)$ or $(v_j^{(t-1)}, z)$ is in C_P. We then label w and z according to the communication in T_P, i.e., if, say, $(v_j^{(t-1)}, w) \in T_P$ then we label w with $v_j^{(t)}$ and v with $v_i^{(t)}$.

$a_i^{(d)}$ **belongs to a simple edge** : $done(i)$ = false and $a_i^{(d)}$ belongs to a simple edge $(a_i^{(d)}, a_i^{(d+1)})$, in which case we look for the w such that $(v_i^{(t-1)}, w) \in T_P$ and we label w with $v_i^{(t)}$. We also set $\phi_P(a_i^{(d)}, 1) = v_i^{(t-1)}$, $\phi_P(a_i^{(d+1)}, 0) = v_i^{(t)}$ and $done(i) =$ true.

$done(i) =$ **true** in which case we look for the w such that $(v_i^{(t-1)}, w) \in T_P$ and we label w with $v_i^{(t)}$.

If the algorithm has processed the entire $2k$ layers of vertices of P and yet $done(i) =$ false for some i, then the algorithm quits (and fails).

Figure 7 illustrates the embedding procedure.

5.3 The mapping is an embedding strategy

The first and second conditions of Definition 4.1 hold directly from the way we choose the embedding ϕ_P. The third condition holds whenever the algorithm stops with a positive successful answer, because we then embed every crossover of G in V_P in a way that uses only honest edges. We later show that the probability of a successful result is high.

Furthermore, the embedding strategy is safe. To see that, fix two executions P and P' of the protocol that use the same sets of vertices, and that agree on all communications over links not in $\boxtimes_P$. As P and P' differ only on crossovers, and the labelling of the vertices at the second layer of

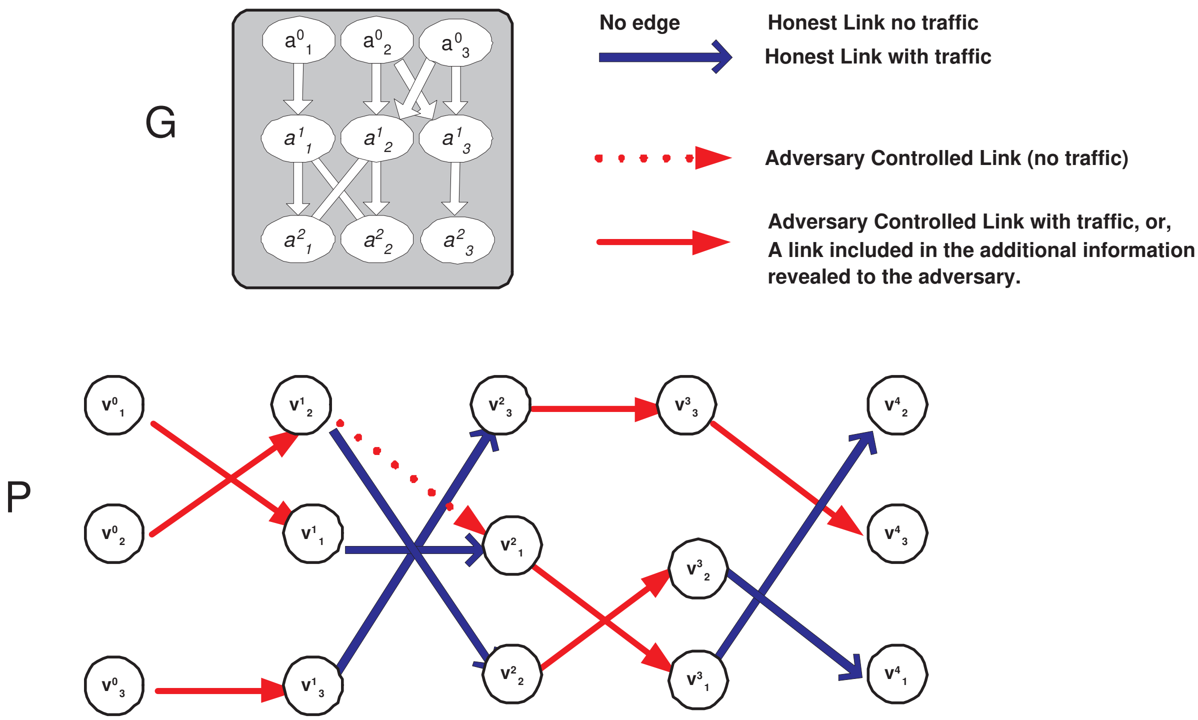

Figure 7: *First notice the labelling of G and see that it is indexed by the layer depth, and respects the edge structure of G.*
Then, look at the labelling of P and notice how it changes when an edge or a crossover in G is successfully embedded into P. The simple edge at the first layer of G was embedded into the second layer of P. The crossover at the first layer of G was not embedded to the second layer of P, because the adversary controls one of the four links of the crossover. It was, however, successfully embedded into the fourth layer of P. Thus, the first layer of G was successfully embedded into the first four layers of P.
Notice also that no embedding is done on odd layers of P.

the crossover depends only on a pre-determined order (not influenced by the actual communication over the crossover), the labelling in P is the same as in P'. It therefore follows that $\phi_P = \phi_{P'}$.

To complete the argument we show that with high probability (over the random coins of the protocol $\mathcal{P}$ from Section 1.5.2) we find all crossovers of $\boxtimes_G$ in P, and the algorithm ends successfully.

Claim. *For every crossover $(a_i^{(d)}, a_j^{(d)}; a_i^{(d+1)}, a_j^{(d+1)}) \in \boxtimes_G$, the probability we do not find it in an execution P of the protocol from Section 1.5.2 is at most $(1-f^4)^k$.*

Proof. Fix a crossover $(a_i^{(d)}, a_j^{(d)}; a_i^{(d+1)}, a_j^{(d+1)}) \in \boxtimes_G$. For every even time step $t = 2kd + \ell$, $2 \le \ell \le 2k$, look at the vertices $v_i^{(t-1)}, v_j^{(t-1)}, v_i^{(t)}, v_j^{(t)}$. These vertices are independent of the vertices in the previous even levels, because in the odd level before, we choose the labelling according to T_P, and in the protocol P next neighbors are chosen at random. It follows by Fact 5.2, that the probability we find the crossover in the level is at least f^4. As different steps are independent, the probability we do not find the crossover in any of the k attempts is at most $(1-f^4)^k$. By the union bound, the probability there exists a crossover which we do not find, is at most $MD \cdot (1-f^4)^k$. □

6 Safe Embeddings and Unlinkability In the No-Prior Information Setting

Intuitively, we can think of a safe embedding strategy as a mechanism that reveals to the adversary whether there was communication or not on all edges not in $\boxtimes_P$, and also that for every $(u_1, u_2; u_3, u_4) \in \boxtimes_P$ the vertices $\{u_1, u_2\}$ communicated with the vertices $\{u_3, u_4\}$. On the other hand, for every crossover $(u_1, u_2; u_3, u_4) \in \boxtimes_P$ the adversary has no knowledge whether u_1 communicated with u_3 and u_2 with u_4, or u_1 with u_4 and u_2 with u_3. Our goal now is to show that if we embed a network that obscures *permutations*, then the adversary gains (almost) no information from his data, *in the no-prior information case*. Somewhat more surprising will be our result that for the above, networks that obscure *inputs* suffice.

6.1 Unlinkability using permutation-obscurant networks

Theorem 6.1. *Assume there exists a depth D network G_M over M inputs that ϵ obscures* permutations*. Assume $\mathcal{P}$ runs in time $T = O(D \log(\frac{DM}{\epsilon}))$. Then in the no-prior information case, $\mathcal{P}$ is $\alpha(N) = O(\epsilon)$ unlinkable.*

Proof. We define the following random variables:

Π : $\Pi(i)$ contains the actual destination of the i'th honest player. The random variable $\Pi = \Pi(1) \dots \Pi(M)$ contains the actual communication pattern between the M honest players and the M destinations. As we are in the no-prior information setting $\Pi = U_{M!}$.

C : C contains all the traffic information the adversary knows. *I.e.*, for every *dishonest* link $(v_i^{(t)}, v_j^{(t+1)})$ it contains the information whether there was traffic on that link during the t'th step or not. C is correlated with Π.

Z : Z contains other information we voluntarily reveal to the adversary. This includes the information whether there was traffic on a link during the t'th step or not, for all links $(v_i^{(t)}, v_j^{(t+1)})$ that do not belong to $\boxtimes_P$. Z is correlated with Π and C.

Fact A.2 tells us that $I(\Pi : C) \leq I(\Pi : C, Z)$. It would therefore suffice to show that $I(\Pi : C, Z) \leq \alpha(N)$.

Let us fix any subset of M honest vertices, and a mapping of the bottom layer of G into the M honest players at the bottom level of P. By Theorem 5.1 the embedding strategy finds (except for ϵ error probability) a safe embedding of G into an execution P of the protocol. From now on we restrict our attention to the event that such a safe embedding exists.

By Definition 4.2, all executions P' of the protocol that use the same set of vertices and agree with P outside $\boxtimes_P$ result in the same embedding. As all the edges that are revealed are outside $\boxtimes_P$, the random variable Z has the same value in both cases. Also, C has the same value in both cases as $\boxtimes_P$ contains only honest edges. Thus, the adversary can not distinguish P from P'. As the a-priori probabilities of the executions P and P' are the same, both are equally likely from the adversary point of view. *I.e.*, any possible communication pattern on $\boxtimes_G$ is equally likely.

Thus, for any values c and z that C and Z might take, any crossover is resolved to be identity with probability half, and a switch with probability half, from the adversary point of view (because all possible communication patterns are equally likely). As G ϵ-obscures *permutation*, it must be the case that the distribution the adversary has over the actual permutation that took place is ϵ close to uniform. $\square$

Notice that here it is natural to use the statistical difference definition of unlinkability (Definition 2.1(2)).

Notice also that our assumption that honest players route messages along a random path is valid only in the no-prior information case. In the prior information case, the distribution on the routes should be such that the a-priori distribution of communication patterns is (the possibly far away from uniform) Π.

6.2 Unlinkability using vertex-obscurant networks

When we use networks that obscure inputs (rather than obscure permutations) the guarantee we get is that for every honest player i, the adversary can not tell where player i sent his message. What we seek is a guarantee that the adversary learns no information at all (not even on subsets of players). Amazingly, networks that obscure inputs suffice for that as well.

Theorem 6.2. *Assume there exists a depth D network G_M over M inputs that ϵ obscures* inputs*. Assume $\mathcal{P}$ runs in time $T = O(D\log(\frac{DM}{\epsilon}))$. Then in the no prior-information case, $\mathcal{P}$ is $\alpha(N) = O(M^3 T\sqrt{\epsilon})$ unlinkable.*

The key to understanding why this happens is the information theoretic chain rule

$$I(CZ : \Pi_1 \ldots \Pi_n) \quad = \quad I(CZ : \Pi_1) + \ldots + I(CZ : \Pi_n | \Pi_1 \ldots \Pi_{n-1}). (1)$$

presented in the Appendix. The chain rule breaks the total amount of information the adversary has about Π to little pieces that are the information that the adversary has regarding specific players. The catch is that we have to bound the amount of information the adversary has regarding player i, *even given complete information about players* $1, \ldots, i-1$. However, in our case, giving complete information about k players just reduces the problem from a problem about M honest players to a problem about $M-k$ honest players. Thus, even with complete knowledge about the first $i-1$ players, the adversary does not gain much information about the i'th player. We now give a formal proof.

Proof. Let H be the set of M honest players, and let us fix an arbitrary mapping of the bottom layer of G_M to H. Let P be an execution of the protocol. With probability at least $1 - MD(1-f^4)^k \geq 1 - \frac{\epsilon}{M}$ there is a safe embedding of G to P, extending the mapping of the bottom layer of G. ¿From now on we restrict our attention to the event that such a safe embedding exists.

As with the proof of Theorem 6.1 this implies that for all values c and z the random variables C, Z respectively get, $(\Pi(1) | C = c, Z = z)$ is ϵ close to uniform, i.e., $\mid (\Pi(1) \mid C = c, Z = z) \; - U_M \mid_1 \; \leq \; \epsilon$. By Lemma 2.1 we see that

$$I(\;\Pi(1) : C) \;\leq\; I(\;\Pi(1) : CZ) \quad\leq\quad log(|\Lambda_N|) \cdot \sqrt{\epsilon} \;\leq\; O(TM^2\sqrt{\epsilon})$$

Where the last inequality is because $log(|\Lambda_N|) = O(TM^2)$.

The crux of our argument is that giving the knowledge of $\Pi(1)$, and in fact even the information of the whole route the message of that player took, just reduces the number of honest players by one, and we are back to the same question, with just one honest player less. Similarly, even given the knowledge of the actions of the first $j-1$ players, and an arbitrary mapping of the bottom layer of $G_{M-(j-1)}$ into the remaining honest players, we can still (with probability at least $1-\frac{\epsilon}{M}$) find a safe embedding of the obscurant network $G_{M-(j-1)}$ into the first T layers of P [3]. Thus, by the above argument,

$$I(C : \Pi(j) \mid \Pi(j-1), \ldots, \Pi(1)) \quad \leq \quad O(TM^2\sqrt{\epsilon}) \tag{2}$$

Thus, by Fact A.1,

$$\begin{aligned} & I(C : \Pi) \\ = \;& I(C : \Pi(1)) + I(C : \Pi(2)|\Pi(1)) + \ldots + I(C : \Pi(M)|\Pi(1), \ldots, \Pi(M-1)) \\ \leq \;& O(MTM^2\sqrt{\epsilon}) \;=\; \alpha(N) \end{aligned}$$

□

Notice that in this proof we work with both the statistical difference definition (Definition 2.1(2)) and the mutual information definition (Definition 2.1(3)) and we use the conversion between them (Lemma 2.1).

We also remark that we look for an embedding of M different graphs $G_1, \ldots, G_M$, with M different sets of honest players, into *the same* T levels of P. We do not claim that there is one embedding that is good for all the M cases (and probably there is not one such embedding). However, the fact that for each of the M possibilities there is a good embedding, implies that Equation (2) holds for all the M possibilities for i, and so by the chain rule the result follows.

Similarly, the adversary auxiliary data Z_i used in each of the M embeddings is different, and it may happen (and probably happens) that the union of the M different auxiliary data Z_i, gives complete information on the traffic. Never the less, the argument goes through because while we get an upper bound on $I(CZ_i : \Pi(j) \mid \Pi(j-1), \ldots, \Pi(1))$ we only use it to imply the weaker upper bound on $I(C : \Pi(j) \mid \Pi(j-1), \ldots, \Pi(1))$.

6.3 Parameters

To prove Theorem 1.1 in the no-prior information case, we choose parameters. Say we want ϵ error. We choose $\epsilon' = (\frac{\epsilon}{N^4})^3$. By Lemma 3.2 we have a network that ϵ'-obscures inputs, with depth $D = O(\log(\frac{M}{\epsilon'}) \cdot \log(M)) = O(\log(\frac{N}{\epsilon}) \cdot \log(N))$. By Theorem 6.2, $T = O(D\log(\frac{DM}{\epsilon'})) =$

[3]Notice that all the M embeddings (for $j = 1, \ldots, M$) are done into the *same* T layers of P.

$O(\log^2(\frac{N}{\epsilon})\log N)$. The error that we get is $\alpha(N) = O(TM^3\sqrt{\epsilon'}) = O(N^4\epsilon'^{1/3}) = \epsilon$ as desired.

7 Unlinkability With Prior Information

We now deal with the general case where the adversary knows that the actual communication that took place has a-priori distribution Π. We note that priors might have extremely low probability in a uniform world. *E.g.*, people tend to communicate with their relatives, an event that has extremely low probability in a uniform world.

One natural attempt for dealing with priors is using Bayes formula. However, this gives very poor results when the prior has a small probability in a uniform world. To illustrate this, consider a random variables A over a universe Λ of size $M!$, that is α close to uniform, $|A - U_{M!}| \leq \alpha$. It is possible that with probability α, A gets some fixed value $a_0 \in \Lambda$ and is uniform otherwise. If so, then conditioning on being in some small subset S that contains a_0 we get:

$$\Pr(A = a_0 | A \in S) \quad = \quad \frac{\Pr(A = a_0)}{\Pr(A \in S)} \approx \frac{\alpha}{\alpha + |S|/|\Lambda|}$$

and so when $|S| < \alpha|\Lambda|$, $(A|S)$ is very far way from $(U|S)$ which is uniform on S. This is bad news for us because, as we discussed before, in our case the prior knowledge can be focused on a negligible fraction of all possible permutations.

7.1 The solution

The reason we can solve the above problem is that in our case the quantifiers are essentially reversed. *I.e.*, the adversary has to choose and fix a prior, and then the players make their random choices, effectively choosing a random network.

The way we technically show our protocol works is by concentrating on the *middle* layer. This is intuitively natural because the adversary knows the permutation at the beginning, and has partial information about the final permutation (given by the prior), but the middle layer seems to be masked by the random choices made throughout the protocol. We let $\Pi^{(T/2)}$ be the random variable whose value is the actual permutation that took place between the first and middle layer. We show that even in the prior knowledge scenario the adversary does not learn much about the middle layer, *i.e.*, that $I(C : \Pi^{(T/2)}) \leq \alpha$. To show that we give the adversary additional information so that to make the information flow only in one direction. Details follow.

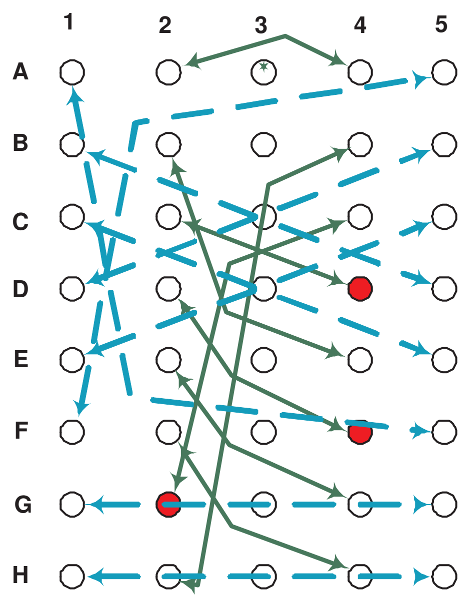

Figure 8: *The figure shows how the vertices of the protocol network in Figure 4 are matched to each other. We show that even with this additional information the adversary does not know much about the middle layer. The reader is encouraged to verify this matching based on the information given in Figure 4.*

Lemma 7.1. *Let* Π *be an arbitrary distribution. Suppose we run the protocol for* $T = \Omega(\log(M)\log^2(\frac{M}{\alpha}))$ *steps. Then* $I(C : \Pi^{(T/2)}) \leq \alpha$.

Proof. We say a vertex $v^{(t)}$ from the t'th layer is associated with a vertex $w^{(T-t)}$ from the $T-t$'th layer, if the message that $v^{(t)}$ forwards eventually arrives at $w^{(T-t)}$. We also say the link (w, v) is associated with the link (v', w') if w is associated with w', and v is associated with v'.

For the proof, we give the adversary the extra knowledge about which vertex at level t is associated with which vertex at level $T-t$, for every $0 \leq t \leq \frac{T}{2}$, and we prove that even with that additional information the adversary does not know much about the middle layer. In Figure 8 we see all the additional information that is given to the adversary for the protocol network of Figure 4. We see that under this additional information the adversary gets to see M players playing our protocol for $T/2$ steps, and where a link $(v^{(t)}, v^{(t+1)})$ is honest iff both the link $(v^{(t)}, v^{(t+1)})$ and its associated link are honest. An example of such a folding is given in Figure 9, where the network of Figure 4 is shown folded.

Thus, the only difference from the case of no prior knowledge is that now the probability each link is honest is f^2 rather than f. We therefore can use Theorem 6.2 for no prior-knowledge (see Section 6.3) and conclude that

$I(C : \Pi^{(T/2)}) \le \alpha$ as desired. $\square$

To complete the proof we show that since the adversary gains very little information about the middle layer, it must be the case that the adversary does not gain much information about the last layer. Formally,

Lemma 7.2. *$I(C' : \Pi^{(T)}) \le I(C' : \Pi^{(T/2)})$.*

Proof. We represent the random variable C that contains the communication the adversary sees as $C = (C_1, C_2)$ where C_1 is the communication seen throughout the first $T/2$ steps, and C_2 is the communication seen throughout the last $T/2$ steps.

We look at $I(\Pi^{(T/2)} : C_1|C_2)$ and define a probabilistic function $f(\sigma, c_2)$ that given $\sigma \in \Pi^{(T/2)}$ and $c_2 \in C_2$ chooses the permutation π with probability $\Pr(\Pi^{(T)} = \pi \mid \Pi^{(T/2)} = \sigma \wedge C_2 = c_2)$. By the data-processing inequality, Fact A.5,

$$I(f(\Pi^{(T/2)}, C_2) : C_1|C_2) \quad \le \quad I(\Pi^{(T/2)} : C_1|C_2) \tag{3}$$

However,

$$I(\Pi^{(T)} : C_1|C_2) \quad = \quad I(f(\Pi^{(T/2)}, C_2) : C_1|C_2) \tag{4}$$

because one way to think of the four correlated random variables $\Pi^{(T/2)}$, $\Pi^{(T)}$, C_1, C_2 is that we first pick $\pi^{(t/2)}, c_1, c_2$ according to the correlated distributions $\Pi^{(T/2)}, C_1, C_2$, and then pick $\pi^{(t)}$ according to the distribution

$$\begin{aligned}(\Pi^{(t)}|\Pi^{(T/2)} = \pi^{(t/2)}, C_1 = c_1, C_2 = c_2) \quad &= \\ (\Pi^{(t)}|\Pi^{(T/2)} = \pi^{(t/2)}, C_2 = c_2) \quad &= \quad f(\pi^{(t/2)}, c_2).\end{aligned}$$

Together we see that

$$\begin{aligned}I(\Pi^{(T)} : C_1, C_2) \quad &= \quad I(\Pi^{(T)} : C_2) + I(\Pi^{(T)} : C_1|C_2) \\ &= \quad I(\Pi^{(T)} : C_1|C_2) \\ &= \quad I(f(\Pi^{(T/2)}, C_2) : C_1|C_2) \\ &\le \quad I(\Pi^{(T/2)} : C_1|C_2) \\ &\le \quad I(\Pi^{(T/2)} : C_1, C_2)\end{aligned}$$

The first and fifth lines are applications of the chain rule, Fact A.1. The third line is Equation (4) and the fourth is Equation (3). To see the second equality, notice that $(C_2|\Pi^{(T)} = \pi)$ is the same distribution for all permutations π that are valid values of $\Pi^{(T)}$. This is because we can think

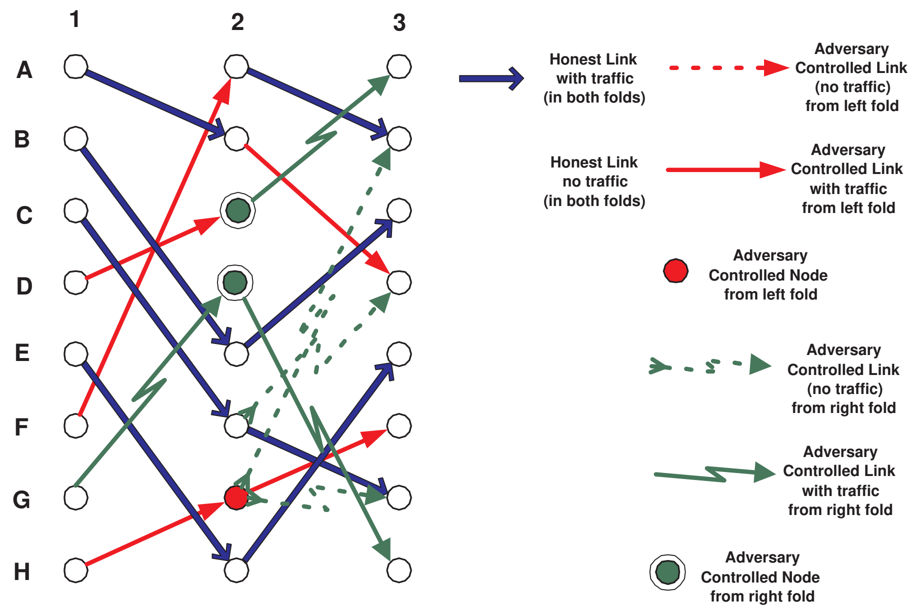

Figure 9: *The figure shows the folding of the protocol network in Figure 4 around its middle layer. In general we also reveal all the even layers. However, to keep the example small, the example shows folding when even layers are not revealed.*
The figure shows the first three layers of the protocol network of Figure 4, combined with additional adversary knowledge obtained from the folding. E.g.,
1. The traffic along the link from $(G, 1)$ to $(D, 2)$ was deduced from the adversary controlled link with traffic from $(F, 4)$ to $(G, 5)$ in Figure 4, when taken along with the mapping $(G, 5) \mapsto (G, 1)$ and $(F, 4) \mapsto (D, 2)$ that we reveal to the adversary during the folding (Figure 8).
2. Adversary knowledge of links with no traffic is also deduced from the folding. E.g., the fact that no traffic goes along the link from $(2, G)$ to $(3, B)$ was deduced from the adversary controlled link with no traffic from $(4, C)$ to $(3, B)$ in Figure 4, when considered along with the folding information of Figure 8.

of the protocol as if the players first pick $\pi \in \Pi^{(T)}$, then pick the top $T-1$ levels at random, and then complete the first layer to implement π. Thus, $I(\Pi^{(T)} : C_2) = 0$. $\square$

This completes our proof for Theorem 1.1 for the general, prior-information case.

7.2 A remark

A few words of caution are required here. While we prove $I(C : \Pi^{(T)}) \leq I(C : \Pi^{(T/2)})$, it is not true that $I(C : \Pi^{(T-1)}) \leq I(C : \Pi^{(T/2)})$. To see that, suppose prior information tells us that $\Pi^{(T)}$ is concentrated on the identity permutation. Then:

- $\Pi^{(T-1)}$ is the uniform distribution over all permutations, but knowing C and $\Pi^{(T)} = id$ usually reveals a lot of information about $\Pi^{(T-1)}$, and so $I(\Pi^{(T-1)} : C)$ is likely to be very large.
- We just proved that $I(\Pi^{(T/2)} : C)$ is very small.

and so it is quite possible that $I(\Pi^{(T-1)} : C) \geq I(\Pi^{(T/2)} : C)$, and in particular $I(\Pi^{(t)} : C)$ is not monotone in t.

This brings us back to the intuition that for an adversary with prior information, the information the adversary sees flows both from bottom up (because the adversary knows who initiates messages, and follows whatever links he can), and from top down (because the adversary has some partial information about who sent who a message, and he follows links from top down).

One can then wonder what in our proof applies to $\Pi^{(T)}$ but does not apply to $\Pi^{(T-1)}$. Going over the proof one finds out that the equation $I(\Pi^{(T)} : C_2) = 0$ does not hold for $\Pi^{(T-1)}$, exactly for the reasons stated above. As before, say $\Pi^{(T)}$ is concentrated on the identity permutation. If we pick $\pi \in \Pi^{(T-1)}$ we can indeed choose the levels *below* it uniformly at random (all except the last one), but we have to also complete the levels above it so that we get the constraints imposed by $\Pi^{(T)}$ (in our case we have to make it the identity permutation, so we are forced to choose π^{-1}). It therefore follows that the distribution $(C_2 | \pi^{(T-1)} = \pi)$ highly depends on π, and so the mutual information might be high.

We believe the proof clearly demonstrates the advantages one gets when quantifying anonymity using information theoretic tools.

8 Extensions

8.1 The Asynchronous Setting

We now describe how to adapt our protocol to the asynchronous setting. Recall that all players have clocks, and all the clocks are within $\Delta_{accuracy}$ time of some global time. Also, the system has a time bound Δ_{bound} on message delivery. We fix some arbitrary global time t_0. The asynchronous protocol specifies that a player can send a message only at times $t_0 + i\Delta$, for $i \in Z$ and $\Delta = 2\Delta_{accuracy} + \Delta_{bound}$. The players then run the above protocol as if their clocks are truly synchronized, and each step takes Δ time.

Let us denote $t^{(i)} = t_0 + i\Delta$. One immediate property is that if M arbitrary players $A_1, \ldots, A_M$ send a message on their local time $t^{(i)}$ to arbitrary M players $B_1, \ldots, B_M$, and players B forward the messages they receive on their next possible time step, then all these messages will be sent in local time $i + 1$. This is regardless of the difference between local times of the different players, and the differences in delivery time. It turns out that this is all we need from our system, thus we get that Theorem 1.1 holds also in the asynchronous model.

8.2 Real Adversaries

The results of this paper remain unchanged even with respect to an adversary model that allows the adversary to eavesdrop traffic on any honest link, 99% of the time, with the caveat that on a random 1% of the time, he does not wiretap the line. This seems a not-unreasonable model for communications in peer to peer networks over the Internet where (at least some) pairs of peers can communicate (at least some of the time) without being recorded.

9 Summary, Later Future Work

9.1 Summary

This paper has dealt mainly with proving that Chaum's protocol is anonymous against adaptive adversaries. We now recall the similarities and differences between our attack model and Chaum's attack model.

Are all links under adversary control? : The main difference between the two attack models is that in Chaum's attack model *no* communication link is honest, while in our attack model we allow 99% of the communication links to be controlled by the adversary, when the remaining 1% of the links are honest. The adversary can choose which of the links are under his control, but cannot change this set dynamically.

Time and time bound : There are several timing attacks on Chaum's system, discussed in [Ray00, RSG98]. Fixes involve a time bound. We include that time bound as an assumption on the system.

Malicious players : Chaum allows malicious nodes that may delete messages. Chaum's handling of malicious nodes, however, is basically the ability to "incriminate" a mix that "failed to properly process an item". We can also supply the same level of incrimination. However, in Section 9.3 we briefly show that this by itself does not guarantee unlinkability, neither in our system, nor in Chaum's system. In fact, Chaum's system does not even stand an adaptive adversary (see, *e.g.*, [Ray00], Sec 3.2).

The resulting protocol is as efficient as Chaum's, and in particular has small message overload (as opposed to many other protocols in the area).

We believe the proof technique has it's own merits:

- We give clear and simple definitions for anonymity. The equivalence we prove between the statistical difference definition and the mutual information definition, proved useful for us, and we believe it will also find other applications.
- We introduced the notion of embeddings (and safe embeddings), and showed how networks that obscure inputs and permutations can be used to achieve anonymity. We believe this is a natural and powerful approach for proving anonymity. It also demonstrates the power of introducing information theory to the field.
- Prior information was usually ignored in previous work. A surprising result is that anonymity can be achieved under a scenario in which the adversary has a lot of information regarding communication patterns. We are not aware of any other work achieving that.

9.2 Later Work

Gomulkiewicz, Klonowski and Kutylowski [GKK04] use our techniques together with path coupling to show that $O(\log(\frac{n}{\epsilon}))$ onion steps suffice in our protocol. The bound is of course tight.

9.3 Future Work

Incomplete network graph : We assumed each node in the system has information about the name, address and public key of the rest of the nodes in the system. This assumption is hardly achievable in real world networks such as the Internet because of their size and unstructured nature. Creating a provable anonymous protocol for incomplete graphs is needed.

It is quite easy to prove the our protocol does not, and can not in its current form, achieve anonymity if the network graph has low degree and the adversary knows the structure of the graph.

Malicious behavior : Malicious adversaries are arbitrary, and in particular can inject, modify or delete messages. The security definition against malicious adversaries is identical to the non-malicious case, requiring low mutual information between collected data and the actual communication pattern. Achieving this in the malicious setting is much more complicated, however.

The ability to "incriminate" a mix that failed to properly process an item, does not, by itself, guarantee security. Say, for example, that M honest players send messages to M other players, and that K malicious players at the first level delete the messages sent to them. If the protocol goes on the adversary then learns how to associate as sets, the set of $M - K$ senders that their messages went through, and the set of $M - K$ receivers, thus learning quite a few bits of information.

A few possible solutions have been proposed in the past (see, *e.g.*, [RS93]). They include shutting down the entire network when such a behavior is detected, and other similar drastic measures. These types of solutions are clearly not practical, and we see this direction of research as being of major importance.

Multi shot game : Running our protocol many times does not increase the mutual information much. It, however, allows the adversary to analyze the *public* senders/recievers data, and this public data itself may reveal a lot of information.

It would be nice to consider a model where the sender/recieive data is not public (unless the sending/recieving node is malicious), and give definitions and protocols that guarantee anonymity even taking the different communication patterns into account.

Network management : Our protocol explicitly assumes that the network on which it runs is static (or at least does not dynamically change). This is not the case (yet again) for real world applications. Dealing with dynamic networks is an important open problem.

References

[Abe99] M. Abe. Mix-networks on permutation networks. In *Proceedings of the Annual Asiacrypt Conference (AsiaCrypt)*, pages 258–273, 1999.

[AH01] M. Abe and F. Hoshino. Remarks on mix-network based on permutation networks. In *International Workshop on Practice and Theory in Public Key Cryptography*, volume 1992, pages 317–324. Lecture Notes in Computer Science, 2001.

[Alo01] N. Alon. Testing subgraphs in large graphs. In *Proceedings of the Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 434–439, 2001.

[Ano] The anonymizer. http://anonymizer.com.

[BD03] A. Beimel and D. Dolev. Buses for anonymous message delivery. *Journal of Cryptology*, 16:25–39, 2003.

[Cha79] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. Thesis (M.S. in Computer Science), University of California, Berkeley, Berkeley, CA, USA, 1979.

[Cha81] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

[Cha88] D. Chaum. The Dining Cryptographers Problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1(1):65–75, 1988.

[CK00] A. Czumaj and M. Kutylowski. Delayed path coupling and generating random permutations. *Random Structures and Algorithms*, 17(3–4):238–259, 2000.

[CKKL99] A. Czumaj, P. Kanarek, M. Kutyłowski, and K. Loryś. Delayed path coupling and generating random permutations via distributed stochastic processes. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 271–280, 1999.

[CT91] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991.

[Dan03] G. Danezis. *Statistical Disclosure Attacks: Traffic Confirmation in Open Environments*, pages 421–426. 2003.

[DFM00] R. Dingledine, M. J. Freedman, and D. Molnar. The free haven project: Distributed anonymous storage service. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *Lecture Notes in Computer Science*. Springer, 2000.

[EFG] Anonymous remailer information. http://anon.efga.org/Remailers.

[Fed00] H. Federrath, editor. volume 2009 of *Lecture Notes in Computer Science*. Springer, 2000.

[FH] The free haven project. http://www.freehaven.net.

[GKK04] M. Gomulkiewicz, M. Klonowski, and M. Kutylowski. Provable unlinkability against traffic analysis already after O(log(n)) steps! In *International Workshop on Information Security*, pages 354–366, 2004.

[GRS96] D. M. Goldschlag, M. G. Reed, and P. F. Syverson. Hiding routing information. In *Information Hiding*, pages 137–150, 1996.

[Mor05] B. Morris. The mixing time of the thorp shuffle. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*, pages 403–412, 2005.

[MP01] D. Malkhi and E. Pavlov. Anonymity without 'cryptography' (extended abstract). In *Financial Cryptography (FC)*, pages 117–135, 2001.

[NC00] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge, 2000.

[PK00] A. Pfitzmann and M. Köhntopp. Anonymity, unobservability, and pseudonymity — A proposal for terminology. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *Lecture Notes in Computer Science*, pages 1–9. Springer, 2000.

[Ray00] J. Raymond. Traffic analysis: Protocols, attacks, design issues, and open problems. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *Lecture Notes in Computer Science*, pages 10–29. Springer, 2000.

[RR98] M. K. Reiter and A. D. Rubin. Crowds: anonymity for Web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[RS93] C. Rackoff and D. R. Simon. Cryptographic defense against traffic analysis. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*, pages 672–681, 1993.

[RSG98] M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, 16(4):482–494, 1998.

[SGR97] P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous connections and onion routing. In *IEEE Symposium on Security and Privacy*, pages 44–54, 1997.

A Some Information Theory Background

A.1 Notations

A discrete distribution D over Λ is a function $D : \Lambda \rightarrow [0..1]$ such that $\sum_{x\in\Lambda} D(x) = 1$. For $S \subseteq \Lambda$ we denote $D(S) = \sum_{s\in S} D(s)$. We identify a random variable A that takes values from Λ with the distribution A such that $A(x)$ is the probability the random variable A takes value x.

We measure distance between distributions defined over Λ with the ℓ_1 norm (also known as the variational distance):

$$|D_1 - D_2|_1 \stackrel{\text{def}}{=} \sum_{x\in\Lambda} |D_1(x) - D_2(x)| = 2 \max_{S\subseteq\Lambda} \; (\; D_1(S) - D_2(S) \;)$$

Let A and B be two events. We denote by $A \otimes B$ their product distribution, *i.e.*,

$$\Pr(A \otimes B = (a, b)) \quad = \quad \Pr(A = a) \cdot \Pr(B = b)$$

and by (A, B) their joint distribution, *i.e.*,

$$\Pr((A, B) = (a, b)) \quad = \quad \Pr(A = a \wedge B = b)$$

A.2 Some Information Theory

The entropy of a distribution D is

$$H(D) \quad \stackrel{\text{def}}{=} \quad \sum_{x\in\Lambda} D(x) \log(\frac{1}{D(x)})$$

The mutual information function is

$$I(A : B) \quad \stackrel{\text{def}}{=} \quad H(A) + H(B) - H(A, B)$$

where

$$H(AB) = \sum_{a\in A, b\in B} \Pr((A, B) = (a, b)) \log(\frac{1}{\Pr((A, B) = (a, b))})$$

The mutual information function can also be expressed as

$$I(A : B) \quad = \quad H(A) - H(A|B)$$

where

$$H(A|B) \quad \stackrel{\text{def}}{=} \quad \mathbb{E}_{b\in B} H(A|B = b)$$

We also define

$$I(X : Y|Z) \quad \stackrel{\text{def}}{=} \quad H(X|Z) + H(Y|Z) - H(X, Y|Z)$$

The mutual information function respects the following chain rule:

Fact A.1. *[CT91]* $I(A : B_1 \dots B_n) = I(A : B_1) + I(A : B_2|B_1) + \dots + I(A : B_n \mid B_1 \dots B_{n-1})$

Also, the mutual information function is monotone in the following sense:

Fact A.2. *[CT91, NC00] For every random variables A, B and C, $I(A : BC) \geq I(A : B)$.*

The entropy function, and hence the mutual information function, are continuous functions. This is captured in:

Fact A.3. *([NC00], Box 11.2, page 512) Assume D_1, D_2 are distributed over Λ. Let $\delta \stackrel{def}{=} |D_1 - D_2|_1$ and assume $\delta \leq \frac{1}{e}$. Then*

$$|H(D_1) - H(D_2)| \quad \leq \quad \delta \log(|\Lambda|) + \delta \log(\frac{1}{\delta})$$

As $I(A : B) = H(A) + H(B) - H(AB)$ we see that:

Fact A.4. *Assume (A, B) is distributed over Λ. Let $\delta \stackrel{def}{=} |(A, B) - (A', B')|_1$ and assume $\delta \leq \frac{1}{e}$. Then*

$$|I(A : B) - I(A' : B')| \quad \leq \quad 3 \cdot (\delta \log(|\Lambda|) + \delta \log(\frac{1}{\delta}))$$

One way to think about the mutual information function $I(A : B)$ is that it measures the amount of information A gets to know about B, or saying it differently, how much knowing the value of A affects our knowledge of the distribution B. An important property of the mutual information function is that A can not increase this knowledge without communicating with B. This is captured in the data processing inequality:

Fact A.5. *(The data processing inequality) (see [NC00], section 11.2.4) For every deterministic or probabilistic function f,*

$$I(f(A, C) : B|C) \quad \leq \quad I(A : B|C)$$

We stress that f should be a function of A and C alone. Also, f might be probabilistic, i.e., uses random coins.

Finally, we define the relative entropy function of two random variables A and B distributed over the same domain:

Definition A.1. [CT91, NC00] Let A and B be distributed over the same domain Λ. Their relative entropy is

$$D(A||B) \quad = \quad \sum_{x \in \Lambda} \Pr(A = x) \cdot \log(\frac{\Pr(A = x)}{\Pr(B = x)})$$

The relative entropy is not symmetric, *i.e.*, $D(A||B)$ might be different from $D(B||A)$. The relative entropy is always non-negative, is zero iff $A = B$ and respects the following inequality:

Fact A.6. *[CT91]* $D(A||B) \geq \frac{1}{2\ln 2}|A - B|_1^2$.

A simple fact is that

Fact A.7. *[CT91, NC00]*
$I(A : B) \; = \; D((A, B)||A \otimes B)$.

A.3 An Example

Rackoff and Simon defined anonymity by using the ℓ_1-norm, see Section 2. A technical problem that arises is that the one-norm is not monotone. One advantage in using the mutual information instead, is that the mutual information is monotone.

To illustrate this issue, consider the following experiment: Toss a 0/1 uniform probability coin 4 times, where $R_i, i = 1, \ldots, 4$, is the result of the i'th coin toss, and $S = \sum_{\ell=1}^{4} R_\ell$. We would like to say that an adversary that knows (R_1, R_2, R_3) "knows" more about S than an adversary that knows only (R_1, R_2). One intuitive way of stating this is that the ℓ_1 distance between the distribution known to the adversary and the binomial distribution for S gets bigger when the adversary learns more. However, this is not necessarily true when we consider specific values of these random variables. Imagine that the adversary already knows $R_1 = R_2 = 0$ and learns that $R_3 = 1$.

$$\begin{aligned}
|(S|R_1 = 0, R_2 = 0) - S|_1 &= \sum_{\ell=0}^{4} \left| \begin{array}{l} \mathrm{Prob}(S = \ell) - \\ \mathrm{Prob}((S|R_1 = 0, R_2 = 0) = \ell) \end{array} \right| \\
&= 0.875; \\
|(S|R_1 = 0, R_2 = 0, R_3 = 1) - S|_1 &= \sum_{\ell=0}^{4} \left| \begin{array}{l} \mathrm{Prob}(S = \ell) - \\ \mathrm{Prob}((S|R_1 = 0, R_2 = 0, R_3 = 1) = \ell) \end{array} \right| \\
&= 0.75.
\end{aligned}$$

Thus, the "little birdy" principle, that revealing additional information about the communications can only help the adversary, does not always work with this definition.

An alternative way of stating that the adversary knows more about S with R_3 than without R_3 is to say that $I(S : (R_1, R_2)) \leq I(S : (R_1, R_2, R_3))$. Indeed,

$$
\begin{aligned}
I(S:(R_1,R_2)) &= H(S)+H((R_1,R_2))-H((S,R_1,R_2)) \\
&= \sum_{\ell=0}^{4} \binom{4}{\ell}/2^4 \cdot \log_2\left(2^4/\binom{10}{\ell}\right) \\
&\quad + \log_2(4) - 4\cdot \sum_{\ell=0}^{2} \binom{2}{\ell}/(4\cdot 2^2)\cdot \log_2\left((4\cdot 2^2)/\binom{2}{\ell}\right) \approx 0.53 \\
I(S:(R_1,R_2,R_3)) &= H(S)+H((R_1,R_2,R_3))-H((S,R_1,R_2,R_3)) \\
&= \sum_{\ell=0}^{4} \binom{4}{\ell}/16 \cdot \log_2\left(16/\binom{4}{\ell}\right) \\
&\quad + \log_2(8) - 8\cdot \sum_{\ell=0}^{1} \binom{1}{\ell}/(8\cdot 2)\cdot \log_2\left((8\cdot 2)/\binom{1}{\ell}\right) \approx 1.03
\end{aligned}
$$

Here, the "little birdy" principle works fine. This is always true (Fact A.2, Section A.2).

In fact, what we learn from that is that *on average* the little birdy principle works with ℓ_1 norm. *I.e.*, while some values may pull us to the wrong direction, most values push towards the right direction. In essence, this is the meaning of the equivalence stated in Lemma 2.5.