# Provable Anonymity for Networks of Mixes[⋆]

Marek Klonowski and Mirosław Kutyłowski

Institute of Mathematics and Computer Science, Wrocław University of Technology,
Marek.Klonowski@im.pwr.wroc.pl[⋆⋆⋆]
Miroslaw.Kutylowski@pwr.wroc.pl

**Abstract.** We analyze networks of mixes used for providing untraceable communication. We consider a network consisting of $k$ mixes working in parallel and exchanging the outputs – which is the most natural architecture for composing mixes of a certain size into networks able to mix a larger number of inputs at once.
We prove that after $O(\log k)$ rounds the network considered provides a fair level of privacy protection for any number of messages. No mathematical proof of this kind has been published before. We show that if at least one of server is corrupted we need substantially more rounds to meet the same requirements of privacy protection.

*Keywords: anonymity, mix network, Markov chain, rapid mixing, coupling*

## 1 Introduction

Providing anonymity becomes one of the key security problems of electronic communication today. There growing dangers both for the private sphere, business communication and information security on a national level.

Many anonymity systems have been proposed - for a collection of papers see [7]. Most anonymity schemes are based on a MIX - a cryptographic primitive introduced by David Chaum [8].

### 1.1 Mixes

A MIX-server processes many encrypted messages at once: it recodes them cryptographically and outputs the re-coded messages in a random order. The purpose of these operations is that no relation between input and output can be established by an adversary that can see the input and the output of a mix.

For instance, a mix described in [8] works as follows (see Fig.1): assume that users $1, 2, \ldots, n$ want to publish anonymously messages $m_1, m_2, m_3, \ldots, m_n$. The users submit their messages to the mix encrypted with the public key $k$ of the mix server: $E_k(m_1), E_k(m_2), E_k(m_3) \ldots, E_k(m_n)$. The mix server decrypts them with its private key, chooses a permutation $\pi$ uniformly at random, and outputs $m_{\pi(1)}, \ldots, m_{\pi(n)}$.

[⋆⋆⋆] contact author

[⋆] preliminary version, accepted for Information Hiding'2005, to be published in LNCS, Springer Verlag, partially supported by KBN

In fact, some additional precautions are necessary: encryption scheme should be probabilistic, duplicates should be removed … . Necessary properties may be implemented in many ways – for example by slightly modified onions [14] or Universal Re-Encryption [9]. For more details see [8] and other papers available on [7].

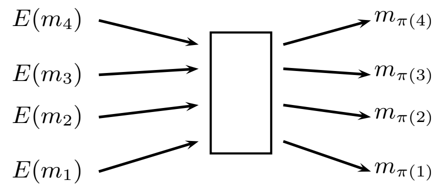

**Fig. 1.** Batch of messages processed by single MIX-server by using random permutation $\pi$

For a proper design, as long as the cryptosystem applied has not been broken, a mix server provides perfect anonymity – after entering the mix messages become indistinguishable. Unfortunately, there are serious drawbacks of this solution. First, we have to trust the (administrator of the) server. There are also scalability problems, since every participant has to use the same server. Of course, this is unrealistic. Also, any server failure (random or caused by an adversary - a “DoS attack”) has severe consequences.

## 1.2 Networks of Mixes

In order to avoid the problems mentioned, many authors propose mixing structures consisting of many MIX-servers interconnected, called *MIXing networks*. The messages entering a network of mixes must visit the MIX-servers in an appropriate order to get to the final destination. At each MIX-server visited a message is recoded appropriately. Of course, encoding scheme must take into account the route of a message.

If a user may distrust certain MIX-servers, then it is reasonable to apply a cascade of mixes: if $k$ MIX-servers are available, then they are pipelined so that the output of server $i$ is the input of server $i + 1$. However, if a mix cannot process all messages at once, a natural solution is a “parallel mix cascade” (see Fig. 2): let us assume that we have MIX–servers $S_1$, $S_2$, …, $S_k$ ($k > 1$) and $n$ messages to be mixed. The protocol consists of $T$ phases (parameter $T$ has to be chosen sufficiently large). During a round $n/k$ messages are submitted to each MIX-server. Each server recodes the messages submitted, permutes the results at random, and sends $n/k^2$ messages to the each server. In particular, $n/k^2$ messages remain at this server.

Details of this protocol may differ slightly in various proposals. For instance, the next MIX-server can be chosen uniformly at random, independently from other servers, instead of directing a fixed fraction to each server. For the rest of the paper we shall consider the former scenario.

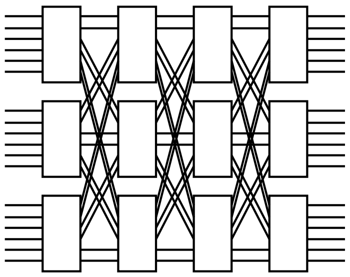

**Fig. 2.** Parallel MIX-cascade with $n = 18$ messages and $k = 3$ mix servers during $T = 4$ steps

## 2 Problem Statement - Anonymity Guarantees

Perfect mixing by the MIX-servers in a MIXing network do not guarantee that all messages get mixed properly. An extreme case of this problem is when each mix gets two inputs at a time – then we have to do with a switching network where switches are set at random. In this case designing a good architecture with quality guarantees remains an unsolved problem. An existing solution [5] with a provable level of anonymity has polylogarithmic depth (i.e., the number of rounds $T$ is a bounded by a polynom in $\log n$, where $n$ is the total number of messages processed), but the exponent is too high for any practical application. However, the situation we are dealing with is not that extreme – we are interested in the case that $k \ll n$.

The general strategy is as follows: in order to permute $n$ messages at random we split the messages into $k$ groups, permute at random each group separately, re-arrange the groups, permute at random each group, re-arrange the groups, and so on. The main question here is how to arrange the groups. Is the parallel mix cascade the best possible architecture? How many rounds are necessary until we approach a random permutation of (encoded) messages?

Still there is no general answer to the question how to permute $n$ elements at random if we have components that may permute $n/k$ elements in one round. There are at least two general approaches: one represented by the parallel mix cascade, and one in which we arrange growing groups of well mixed elements. The main part of such algorithms is a shuffling procedure that take two groups and merges them.

We assume that there is an adversary trying to trace messages going through a MIX-network. He knows the algorithm but cannot break the cryptographic scheme used. We can also consider a situation that an adversary controls certain mixes and therefore knows the permutations applied by these mixes.

### 2.1 Previous Work

The very first paper introducing MIXes as a tool for enhancing anonymity level was published by David Chaum [8]. He also proposed pipelining of several MIX-servers that form a "MIX-cascade". In [14] Rackoff and Simon presented a very significant extension of Chaum's scheme – in this protocol the route of a message is determined by a sender and the message is encoded in a structure resembling an onion. A similar approach was used later by many authors (e.g. [15, 12]), but progress in estimating the runtime such that a good level of anonymity is achieved, was quite slow. The topology of parallel mix cascade that we consider in this paper was described for instance in [12].

The first paper with an anonymity proof of a mixing protocol was published by Rackoff and Simon [14]. Further results about the same protocol under a different adversary model have been obtained recently [3, 11]. None of these results applies to the situation considered in this paper – the main focus of these papers is on how much information is granted to the adversary through traffic information. In our case the communication is oblivious and does not depend on how the messages are mixed.

### 2.2 New results

We provide a detailed analysis of parallel mix cascades and estimate how many rounds are necessary until the probability distribution over possible mappings between the messages entering the network and those leaving the network becomes close to uniform. It turns out that this number does not depend on the number of messages! In this case the adversary cannot link the decoded messages that leave the network with the messages submitted by the users – the probabilities are close to the case of the uniform probability distribution. This result is given by Theorem 1.

Our result is based on *delayed path coupling* - a technique introduced in [6], which is an extension of path coupling [1]. In fact, there are some traces of the current technical approach in our former paper [10], namely for the case $k = 2$.

### 2.3 Paper Organization

In Section 3 we consider adversary and the definition of anonymity that we use. We discuss very quickly why this definition provides better anonymity guarantees than some other definitions used in the literature. Section 4 is devoted to the main result. We start it with a description of mathematical tools necessary to analyze mix networks. In Section 5 we analyze the situation in which one of the MIX-servers is corrupt.

## 3 Mixing Network and Anonymous Communication

**Adversary Model** There are substantially different models of an adversary trying to establish relation between messages entering a MIX-system and the messages leaving this system. Many papers deal with so–called a *global passive adversary*. This adversary can just observe the traffic on some number of links and servers (in our case the traffic on the links is oblivious, so the traffic on the links is of no use). The most common model assumes that an adversary can eavesdrop all links but none of servers. If a server reveals permutation that it use to an adversary we call it *corrupted* or *dishonest*.

An active adversary can add, remove, replace, duplicate messages at certain nodes or links. This turns out to be very dangerous and there are several papers dealing with these issues – for example [9, 13].

Our main result refers to the model with a global passive adversary with no control over the servers. In Section 5 we consider the case of dishonest servers.

**Anonymity Definition** Anonymity is a vague notion that can be formalized in various ways. Since it is desired to provide some security *guarantees*, we consider one of the strongest notions.

A relation between $n$ messages entering and leaving the whole mixing structure can be described by a permutation of elements $\{1, \ldots, n\}$. An adversary wants to reveal (at least partially) this permutation using information at hand or at least to know that some permutations are much more probable than the others.

Already applying a few mixes the number of possible permutations is quite a big one. So an adversary can consider the permutation of messages $\{1, \ldots, n\}$ as a random variable with some probability distribution $\Pi_t$ depending on the traffic information gained by the adversary. Of course, we would wish that $\Pi_t$ is a uniform distribution $\mu_U$. Unfortunately, in many systems it is impossible due to simple divisibility reasons. In particular, it cannot happen for a parallel MIX cascade for any number of rounds.

On the other hand, stochastic process $\{\Pi_t\}_{t \in \mathbb{N}^+ \cup 0}$ converges to the uniform distribution over $\mathbb{S}_n$ in respect to the metrics *total variation distance*. Recall that for random variables $\Gamma_1$ and $\Gamma_2$ with a finite set of values $\mathcal{Y}$ variation distance between $\Gamma_1$ and $\Gamma_2$ is defined by the formula:

$$\mathrm{TVD}(\Gamma_1, \Gamma_2) = \tfrac{1}{2} \textstyle\sum_{y \in \mathcal{Y}} |\Pr(\Gamma_1 = y) - \Pr(\Gamma_2 = y)| \ .$$

(Sometimes in this paper we use distribution of random variable instead of random variable itself.) We say that the parallel mix cascade provides anonymity after $t_0$ steps if

$$\mathrm{TVD}(\mu_U, \Pi_t) < 1/c$$

or every $\geq t_0$. Parameter $c$ is usually $1/n^{\alpha}$. This definition takes into account not only distribution of a single message, but also correlations between them. This is very important, for instance in the case when mixes are used for electronic voting [10].

The anonymity definition considered is equivalent to a definition based on information theory [3]. Some other papers use yet another substantially weaker definitions (see for example [13]). They might be useful and suffice in concrete situations. On the other hand, since we prove a bound for a strong definition, our result applies also to the weaker ones.

## 4 Coupling Proof of the Main Result

It is intuitively obvious for the parallel mix cascade the distribution of $\Pi_t$ converges to the uniform distribution. We use a standard convergence measure called *mixing time*:

$$\tau_{\mathbf{M}}(\varepsilon) = \min \{T : \forall \pi \in \mathbf{S_n}, \forall t \geq T \ \mathrm{TVD}(\mathcal{L}_\pi(\Pi_t), \mu) \leq \varepsilon\} \ .$$

where $\mathcal{L}_\pi(\Pi_t)$ denotes probability distribution of a random variable $\Pi_t$ under the condition that $X_{t_0} = \pi$ and $\mu$ is the uniform distribution.

In this section we formally prove that $\{\Pi_t\}_{t \in \mathbb{N}}$ converges very quickly to $\mu$. For this purpose we use *delayed path coupling* technique described below.

**Delayed Path Coupling** Delayed Path Coupling [5] is a tool for proving convergence rate of a homogeneous Markov chain. It is an extension of Path Coupling [4] and the generic coupling method. Let us recall it briefly. Let $\mathbf{M} = (\mathcal{Y}_t)_{t\in\mathbb{N}}$ be a discrete-time Markov chain with a finite state space $\mathbf{S}$ that has a unique stationary distribution $\mu$. A *coupling* for a Markov chain $(\mathcal{Y}_t)_{t\in\mathbb{N}}$ is a stochastic process $(Y_t, Y^\star_t)$ on the space $\mathbf{S}\times\mathbf{S}$ such that processes $Y_t$ and $Y^\star_t$ considered separately are faithful copies of $\mathcal{Y}_t$. In other words, $\Pr(\mathcal{Y}_{t+1} = y|\mathcal{Y}_t = x) = \Pr(Y_{t+1} = y|Y_t = x) = \Pr(Y^\star_{t+1} = y|Y^\star_t = x)$ for each $x, y \in \mathbf{S}$.

We assume that there is a metric $\varDelta : \mathbf{S}\times\mathbf{S} \longrightarrow \mathbb{N}$; let $D$ be the largest distance according to metrics $\varDelta$. Further, let

$$\varGamma = \{(Y_{t\delta}, Y^\star_{t\delta}) \in \mathbf{S}\times\mathbf{S} : \varDelta(Y_{t\delta}, Y^\star_{t\delta}) = 1\} \ .$$

Further, we need to assume that for all $(Y_{t\delta}, Y^\star_{t\delta}) \in \mathbf{S}\times\mathbf{S}$, if $\varDelta(Y_{t\delta}, Y^\star_{t\delta}) = r$, then there exist a sequence (a "path") $Y = \varLambda_0, \varLambda_1, \ldots, \varLambda_r = Y^\star$ with $(\varLambda_{i-1}, \varLambda_i) \in \varGamma$ for $0 \le i < r$. No we can formulate the main technical result on delayed path coupling:

**Lemma 1 (Delayed Path Coupling Lemma).** *Assume that there exist a coupling $(Y_{t\delta}, Y^\star_{t\delta})$ for a process $(\mathcal{Y}_t)_{t\in\mathbb{N}}$ such that for some real $\beta < 1$ and positive integer $\delta$ we have $\mathbf{E}[\varDelta(Y_{(t+1)\delta}, Y^\star_{(t+1)\delta})] \le \beta$ for all $(Y_{t\delta}, Y^\star_{t\delta}) \in \varGamma$ and for all $t \in \mathbb{N}$. Then,*

$$\tau_{\mathbf{M}}(\varepsilon) \le \delta \cdot \lceil \ln(D\varepsilon^{-1})/\ln\beta^{-1} \rceil \ .$$

For further details on delayed path coupling see [5,4]. By Lemma 1, in order to estimate the total variation distance between probability distribution describing the state of a process after step $t$ and its stationary distribution it suffices to construct an appropriate coupling. We should stress that processes $Y_t$ and $Y^\star_t$ are usually dependent – constructing a proper dependence between them is the tricky part of the proof.

## 4.1 Main Result

**Theorem 1 (Main Result)** *For a parallel MIX cascade TVD$(\mu_U, \varPi_t) = \frac{1}{n}$ for $t > T$, where $T = \mathcal{O}(\log k)$ and does not depend the number of messages $n$.*

**Preliminaries** At the beginning we observe that $\{\varPi_t\}_{t\in\mathbb{N}}$ is a symmetric and ergodic Markov chain. So the uniform distribution is its unique stationary distribution.

The second key observation is that after mixing in the first step we can confine ourselves to permutations of $n$ balls colored with $k$ different colors - $n/k$ balls of each color. We say that a ball has color $i$, if it is processed by server $i$ in the first step of the protocol. So the process $\{\varPi_t\}_{t\in\mathbb{N}}$ has values in a space of all placements of such balls on $n$ positions. We denote this space by $\mathbf{S}$.

For $y_1, y_2 \in \mathbf{S}$ we define $\varDelta(y_1, y_2)$ to be the minimal number of transpositions necessary to get from state $y_1$ to $y_2$. Of course, $\varDelta$ is a metric and $\max_{y_1,y_2} \varDelta(y_1, y_2) = n(k-1)$.

**Construction of Coupling** Let processes $(\Upsilon_t, \Upsilon_t^\star)$ differ on one position at time $t$. Let $i_0$ and $j_0$ be the positions at which the configurations $\Upsilon_t, \Upsilon_t^\star$ do not match, say the first process has a white ball at position $i_0$ and a black ball at position $j_0$, while for the second process the roles are reversed: a black ball is at $i_0$ and a white ball is at $i_0$.

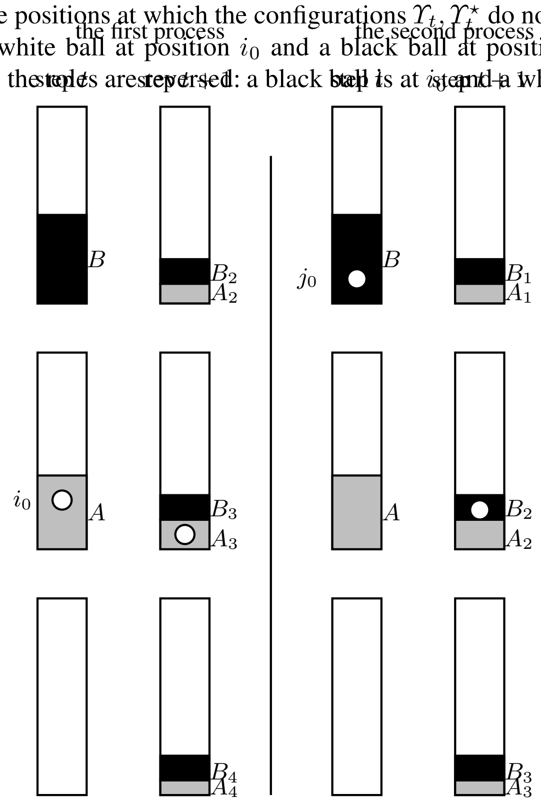

**Fig. 3.** coupling idea

We shall talk about an "extra white ball" for each process. This is the white ball that originally stands at position $i_0$ for the first process and ball at position $j_0$ for the second process. This terminology refers to the fact that if we replace them by black balls, then the states of both processors become the same. In our approach we take into account black balls and the extra white balls of each process. The remaining balls go to the same positions for both processes. Since the remaining positions are left for the black balls and for the extra white balls, the configuration of each process is determined by the placement of the extra white ball on the positions that are left.

Let $S_1$, $S_2 \dots S_k$ denote sets of positions corresponding, respectively to mixes, 1, 2, ..., $k$. Observe that if the extra white balls from both processes are placed in the same server, i.e. $\Upsilon_{t+1}(i_0) \in S_l$ and $\Upsilon_{t+1}(j_0) \in S_l$ for an $l \leq k$, then we can couple the processes. Namely, the second process uses the same permutation $\pi$ for $S_l$ as the first process but composed with a transposition concerning the positions of the extra white balls (the transposition is applied before $\pi$). It is easy to see that this operation does not change the marginal probability distribution of the second process, because $\pi$ has the uniform distribution and so composition of a fixed transposition and of $\pi$ yields a uniformly distributed permutation. Unfortunately, the situation described has a low probability, namely $1/k$. For this reason a more sophisticated strategy is necessary.

For presenting coupling construction we need some notations. Let $i_0 \in S_i$ and $j_0 \in S_j$. Let $A$ and $B$ be the set of positions occupied by the black balls and the extra white balls in $S_i$ and $S_j$. Moreover, let $A_i = \Upsilon(t)(A) \cap S_i$ and $B_i = \Upsilon(t)(B) \cap S_i$. That is, $A_i$ ($B_i$) is the set of positions occupied in step $t+1$ in server $i$ by balls that were at positions from set $A$ ($B$, respectively) at step $t$. Further, let: $a = |A|$ , $a_i = |A_i|$, $b = |B|$, and $b_i = |B_i|$. Let $I_A = \{i \mid \frac{a_i}{a} > \frac{b_i}{b}\}$ and $I_B = \{1 \ldots k\} \setminus I_A$.

**Coupling Definition**

1. We let the first process to place all balls except these starting from positions in set $A \cup B$. The second process places the corresponding balls in exactly the same way.
2. Now the extra white ball (of the first process) from position $i_0$ is placed uniformly at random on one of the positions that are left – i.e. $\Upsilon(t)(A \cup B)$.
   Now let us assume that $\Upsilon_t(i_0) \in S_h$. We define how to place the second extra white ball. Let $U = \sum_{i \in I_B}(\frac{b_i}{b} - \frac{a_i}{a})$. (So $U = \sum_{i \in I_A}(\frac{a_i}{a} - \frac{b_i}{b})$, as well.)
   **Case** $1$ – $h \in I_B$**:** in this case we put the extra white ball of the second process on a position chosen uniformly at random from the set $B_h$.
   **Case** $2$ – $h \in I_A$**:** we toss an asymmetric coin:
   - with probability $\frac{b_h}{b} / \frac{a_h}{a} = \frac{b_h \cdot a}{b \cdot a_h}$ we place the extra white ball of the second process on randomly chosen position from set $B_h$,
   - with probability $1 - \frac{b_h \cdot a}{b \cdot a_h}$, we choose a position at random: we put the extra white ball into set $B_l$ with probability
     $$\frac{(b_l/b) - (a_l/a)}{U}$$
     for $l \in I_B$. The choice of position within $B_l$ is uniform.
3. We are left with black balls only. Each process simply places them at the remaining unoccupied positions.

If the extra white balls fall into the same server at step $t+1$, then in the next step we couple successfully the processes as it was described above.

**Correctness of Coupling** Now we have to show that the procedure described above is a proper coupling - that is the second process has appropriate probability distribution of transitions. Obviously, it is enough to show that the extra white ball from the second process will be placed in each set $B_h$ with probability $\frac{b_h}{b}$. We consider two cases:

**Case** $1$ – $h \in I_B$**:** In this case the extra white ball had to fall into the set $A_i$ and we had a "positive" result of tossing the coin. So

$$\Pr(\Upsilon_t(j_0) \in B_i) = \frac{a_h}{a}\left(\frac{b_h \cdot a}{b \cdot a_h}\right) = \frac{b_h}{b} \ .$$

**Case** $2$ **–** $h \in I_B$**:** In this situation the extra white ball from the second process could be placed in set $B_h$ in two situations. Either the extra white ball from the first process falls into set $A_h$ or it falls into some of the sets $A_i, i \in I_A$. Events leading to these situations are disjoint. So:

$$\Pr( \Upsilon_t(j_0) \in B_h) = \frac{a_h}{a} + \sum_{i \in I_A} \frac{a_i}{a} \cdot \left(1 - \frac{b_i \cdot a}{b \cdot a_i}\right) \cdot \frac{b_h/b - a_h/a}{U} =$$

$$\frac{a_h}{a} + \frac{b_h/b - a_h/a}{U} \cdot \sum_{i \in I_A} \frac{a_i}{a} \left(1 - \frac{b_i}{b} \cdot \frac{a}{a_i}\right) = \frac{a_h}{a} + \frac{b_h/b - a_h/a}{U} \cdot U = \frac{b_h}{b} \; .$$

**Success Probability of Coupling Strategy** Now we estimate probability of the event that the processes get coupled in two steps. As we have seen, it happens at the beginning of step $t+2$ if at the end of step $t+1$ the extra white balls get into the same server.

As before, we consider two cases with respect to the position of the extra white ball of the second process at step $t+1$. Let $\Upsilon_t(i_0) \in S_h$.

**Case** $h \in I_A$**:** the processes get coupled with probability $\frac{b_h \cdot a}{b \cdot a_h}$. So this contributes $\frac{a_h}{a} \cdot \frac{b_h \cdot a}{b \cdot a_h} = \frac{b_h}{b}$ to the overall probability of successful coupling.

**Case** $h \in I_B$**:** the processes get coupled with probability $1$. So this contributes $\frac{a_h}{a}$ to the overall probability of successful coupling.

Hence:

$$\Pr(\Upsilon_{t+2} = \Upsilon^{\star}_{t+2}) = \textstyle\sum_{i \in I_A} \frac{b_i}{b} + \sum_{i \in I_B} \frac{a_i}{a} = \sum_{i=1}^{k} \min\left\{\frac{a_i}{a}, \frac{b_i}{b}\right\} \; .$$

Now our goal is to estimate $\frac{a_i}{a}$ and $\frac{b_i}{b}$

**Lemma 2.** *Starting at any situation in our model after one step each server contains at least* $n/16k^2$ *balls of each color with probability greater then* $1 - \exp(-n/32k^2)$

**Sketch of the proof.** Without loss of generality, we consider the black balls and server $S_1$, only. Let us assume that in the first step we have $x_i$ black balls in the server $S_i$. So $x_1 + x_2 + \ldots + x_k = n/k$. Let $y_i$ be the number of black balls that goes from server $S_i$ in first step to server $S_1$ in the second step. Of course $y_i \leq x_i$. We are interested in estimating $y = y_1 + y_2 + \ldots + y_k$ We consider two cases:

$x_i \leq n/2k^2$ : then we can estimate $y_i$ from below by $x_i$ independent Bernoulli trials each with probability $1/2k$.

$x_i > n/2k^2$ : in this case we can estimate from below $y_i$ by $n/4k^2$ Bernoulli trials each with probability of success $x_i k/2n$.

Indeed, if $x_i > n/2k^2$, then probability of assigning the $j$th position connecting $S_j$ with $S_1$ to a black ball after making decisions about positions 1 through $j-1$ is at least

$$(x_i - (j-1))/(n/k) \geq (x_i - n/4k^2)/(n/k) > (x_i/2)/(n/k) = x_i k/2n \; .$$

W.l.o.g. let us assume that exactly the first $l$ servers belongs to the first category – they have $x_i \leq n/2k^2$ black balls at first step. The remaining $k-l$ servers belong to the second category. Then

$$\begin{aligned} E(y) &\geq (x_1 + \ldots + x_l)\tfrac{1}{2k} + \tfrac{n}{4k^2}\tfrac{x_{l+1}k}{2n} + \ldots + \tfrac{n}{4k^2}\tfrac{x_k k}{2n} \geq \\ &= (x_1 + \ldots + x_k)\tfrac{1}{8k} = \tfrac{n}{8k^2} \ . \end{aligned}$$

We use Chernoff bound in the following form: if $X$ is a sum of independent random variables, then $\Pr(X \leq (1-\delta)\mathrm{E}(X)) \leq \exp(-\delta^2 \mathrm{E}(X))$ for each $0 < \delta < 1$. So for $\delta = 1/2$ and the previous estimations we get that $\Pr(y \leq \frac{n}{16k^2}) \leq \exp(-n/32k^2)$.

Let us recall the following lemma probability theory (see [2]):

**Lemma 3.** *Let assume that we choose $\beta N$ balls at random from set of $\alpha N$ black balls and $(1-\alpha)N$ white balls without replacing. Let $X$ be the number of black balls chosen. Then for $\gamma > 0$*

$$\Pr(|X - \alpha\beta N| > \sqrt{2\beta\gamma N}) \leq 2\exp(-\gamma) \ .$$

Now we can estimate the values $a_i/a$ and $b_i/b$. We use Lemma 3 with the parameters $N = n/k$, $\alpha = ak/n$ and $\beta = 1/k$. We get:

$$\Pr\left(\left|a_i - \tfrac{a}{k}\right| > \sqrt{2\gamma n}/k\right) < 2\exp(-\gamma) \ ,$$

$$\Pr\left(a_i < \tfrac{a}{k} - \sqrt{2\gamma n}/k\right) < 2\exp(-\gamma) \ .$$

So by dividing by $a$ we get

$$\Pr\left(\tfrac{a_i}{a} < \tfrac{1}{k} - \tfrac{\sqrt{2\gamma n}}{ka}\right) < 2\exp(-\gamma) \ .$$

By Lemma 2 applied to $a$ in expression written above and using very rough estimation of probabilities we have for $\gamma = n^{0.4}$:

$$\Pr\left(\frac{a_i}{a} < \frac{1}{k} - \frac{\sqrt{2}}{k}\frac{16}{n^{0.3}}\right) < 2\exp(-n^{0.4}) + \exp(-n/32k^2) \ .$$

Note that analogous formulas hold for $a_i/a$ as well as $b_i/b$ for all $0 < i \leq k$. Thereby

$$\begin{aligned} \Pr(\Upsilon_{t+2} = \Upsilon^\star_{t+2}) &= \textstyle\sum_{i=1}^{k} \min\left\{\tfrac{a_i}{a}, \tfrac{b_i}{b}\right\} \\ &> k\left(\tfrac{1}{k} - \tfrac{\sqrt{2}}{k}\tfrac{16}{n^{0.3}}\right) - 2k(2\exp(-n^{0.4}) + \exp(-n/32k^2)) \ . \end{aligned}$$

For sufficiently large $n$ we can estimate the expression above by $1 - 23/n^{0.3}$.

**Stopping Time** Now we have all factors necessary to evaluate formula of Delayed Path Coupling. Since in our coupling the distance cannot increase we have:

$$\beta = E(\Delta(\Upsilon_{t+2}, \Upsilon^\star_{t+2})) = \Pr(\Upsilon_{t+2} \neq \Upsilon^\star_{t+2}) < 23/n^{0.3}$$

By our construction, $\delta = 3$. Since $D = n(k-1)$ we get finally

$$\tau_{\mathbf{M}}(\varepsilon) \leq 3\left\lceil\frac{\ln n(k-1)\varepsilon^{-1})}{0.3\ln n - \ln 23}\right\rceil \ .$$

So for a standard value $\varepsilon = 1/n$ used in the literature we have $\tau_{\mathbf{M}} = \mathcal{O}(\log k)$, which is $\mathcal{O}(1)$ with respect to the number of messages $n$.

## 5 Dishonest Server Case

In this section we compare the results obtained in the previous section with the case in which at least one server is dishonest (i.e. reveals permutations used to an adversary), and show that we need significantly more steps to achieve the same level of anonymity. Namely, namely this number of steps becomes a function of $n$, which is a significant difference with the previous case.

First observe that if the exact position of at least one message after mixing process is known - i.e. $\widehat{\Pi}_T(i) = j$ for certain $i, j$, then

$$\text{TVD}(\widehat{\Pi}_T, \mu_U) \geq 1 - 1/n \ .$$

(This shows how sensitive is total variation distance as a measure of anonymity.) Obviously, the variation distance considered reaches a minimum if $\widehat{\Pi}_T$ maps all messages (except $i$) uniformly at random on positions $\{1, \ldots, n\} \setminus \{j\}$. In this case

$$\text{TVD}(\widehat{\Pi}_T, \mu_U) = \frac{1}{2}\left((n-1)!\left|\frac{1}{(n-1)!} - \frac{1}{n!}\right| + (n! - (n-1)!)\left|0 - \frac{1}{n!}\right|\right) = 1 - 1/n \ .$$

Now we check that if a dishonest server permanently reveals how it permutes the messages, then with a constant probability the route of some message will be revealed for $T = \Theta(\log\log n/\log k)$ steps. After the first step exactly $n/k^2$ messages remain at the dishonest server. We choose $n/2k^2$ of them and for each of them estimate the chance that it will remain at the dishonest server all the time. In order to analyze a single step we assume that the distinguished messages choose the output position of the mix at random: the first message can choose an arbitrary output position, the next message chooses at random from all output positions of this mix except the one occupied by the first message, the third message chooses among $n/k$-2 positions, and so on. In any case each of the distinguished messages has probability at least $\frac{n/k^2 - n/(2k^2)}{n/k} = 1/2k$ to remain at the dishonest server, hence at least $(1/2k)^T$ within $T$ steps, no matter what happens with the remaining distinguished messages. Hence the probability that at least one message stays at the same dishonest server is grater than

$$1 - \left(1 - \left(\frac{1}{2k}\right)^T\right)^{n/2k^2} \ .$$

So if $T = \Theta(\log\log n/\log k)$, then with a constant probability an adversary can trace the whole route of some message.

## 6 Conclusions and Open Problems

In this paper we have proved that parallel mix cascade provides very high standard of privacy security. We need $\mathcal{O}(1)$ protocol steps if all servers are honest. It is not enough when at least one server is dishonest or corrupted.

We have shown an interesting phenomenon that if all servers of parallel mix cascade are honest, the number of steps necessary to achieve good provable anonymity does not depend on the number of messages, while it is not true if a single mix is dishonest.

It is still open question how many steps we do need for other mix-network topologies and what is the optimal topology.

## References

1. Aldous, D.: *Random Walks of Finite Groups and Rapidly Mixing Markov Chains*. In: Azéma, J., Yor, M. (eds.): Séminare de Probabilités XVII 1981/82. Lecture Notes in Mathematics, Vol. 986. Springer-Verlag, Berlin (1983), 243-297
2. Auletta, V., Caragiannis, I., Kaklamanis, C., Persiano, P.: *Randomized Path Coloring on Binary Trees*. APPROX'2000, LNCS 1913, Springer-Verlag, 60-71
3. Berman, R., Fiat, A., Ta-Shma, A.: *Provable Unlinkability Against Traffic Analysis*. Financial Cryptography 2004, LNCS 3110, Springer-Verlag, 266-280
4. Bubley, B., Dyer, M.: *Path Coupling: A Technique for Proving Rapid Mixing in Markov Chains*. ACM-SIAM FOCS '38, 1997, 223-231
5. Czumaj, A., Kanarek, P., Kutyłowski, M., Loryś, K.: *Switching Networks for Generating Random Permutations*. In: Switching Networks: Recent Advances. Kluwer Academic Publishers, (2001)
6. Czumaj, A., Kutyłowski, M.: *Delayed Path Coupling and Generating Random Permutations*. Random Structures and Algorithms (2000) 17(3-4): 238-259
7. Dingledine, R. Anonymity Bibliography http://freehaven.net/anonbib/
8. Chaum, D.; *Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms*. CACM 24(2) (1981) , 84-88
9. Golle, P., Jakobsson, M., Juels, A., Syverson, P.: *Universal Re-encryption for Mixnets*. RSA-CT'04 LNCS 2964, Springer-Verlag, 163-178
10. Gomułkiewicz, M., Klonowski, M., Kutyłowski, M.: *Rapid Mixing and Security of Chaum's Visual Electronic Voting*, ESORICS'2003, LNCS 2808, Springer-Verlag, 132-145
11. Gomułkiewicz, M., Klonowski, M., Kutyłowski, M.: *Provable Unlinkability Against Traffic Analysis Already After $\mathcal{O}(\log(n))$ Steps!*, Information Security Conference (ISC)'2004, LNCS 3225, Springer-Verlag, 354-366
12. Gülcü, C., Tsudik, G.: *Mixing E-mail with BABEL*. ISOC Symposium on Network and Distributed System Security, IEEE 1996, 2-16
13. Kesdogan, D., Egner, J., Büschkes, R.: *Stop-and-Go-MIXes Providing Probabilistic Anonymity in an Open System*. Information Hiding '98 LNCS 1525, Springer-Verlag, 83-98
14. Rackoff, C., Simon, D.R.: *Cryptographic Defense Against Traffic Analysis*. ACM Symposium on Theory of Computing25 (1993) 672-681
15. Syverson, P. F., Reed, M. G., Goldschlag, D. M.: *Anonymous Connections and Onion Routing*. IEEE Journal on Selected Areas in Communication, 1998, 16(4):482-494