

Does Anonymity Love the Chat Groups?

Marc Roßberger
marc.rossberger@ur.de
University of Regensburg
Regensburg, Germany

Doğan Kesdoğan
kesdogan@ur.de
University of Regensburg
Regensburg, Germany

Abstract

Instant messaging is an integral part of daily communication, and users increasingly seek anonymous options to protect their privacy. Mix networks are a well-established approach for metadata protection, but current designs largely overlook the unique challenges of instant messaging, such as low-latency expectations, bursty interaction patterns, and group chats. In this work, we present the first systematic analysis of mix networks, and anonymization systems in general, for group messaging under realistic conditions. We introduce a novel group-identification attack that exploits co-occurrence patterns of recipients to infer group membership with high accuracy, even against a significantly weaker attacker model than these systems typically assume, a local observer. Our evaluation leverages real-world WhatsApp data and adapts it to better reflect realistic usage scenarios. Finally, we evaluate how defense strategies such as relaxing low latency requirements to allow for more delay options and cover traffic perform against this attack. Our findings reveal critical weaknesses in current designs and provide insights for improving anonymity in instant group messaging.

CCS Concepts

• Security and privacy → Usability in security and privacy; Pseudonymity, anonymity and untraceability; Privacy-preserving protocols.

Keywords

Anonymous Communication; Instant Messaging; Mix Network

ACM Reference Format:

Marc Roßberger and Doğan Kesdoğan. 2026. Does Anonymity Love the Chat Groups?. In *Proceedings of the Sixteenth ACM Conference on Data and Application Security and Privacy (CODASPY '26)*, June 23–25, 2026, Frankfurt am Main, Germany. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3800506.3803513>

1 Introduction

WhatsApp’s mass adoption illustrates both the convenience and the risks of modern messaging platforms. A recent incident showed how a seemingly harmless feature, checking if a phone number is registered, enabled large-scale harvesting of billions of users’ phone numbers, profile pictures, and identifying text [28]. This example highlights how small design decisions can lead to massive privacy breaches. Similarly, even when strong encryption is in place, structural aspects of instant messaging, such as group chats,

can expose sensitive metadata. Unlike direct data leaks, metadata reveals who communicates with whom and when, enabling adversaries to infer social relationships and group structures. Protecting this metadata is therefore critical for achieving meaningful privacy in modern messaging systems, especially for vulnerable groups such as journalists or activists.

Mix networks offer a well-established approach to anonymizing communication relationships by shuffling messages from different users across multiple nodes. However, prior research on mix networks has focused almost exclusively on one-on-one communication. Modern instant messaging platforms, however, are also shaped by group chats, which introduce unique structural and behavioral characteristics. Group messaging amplifies metadata leakage because multiple recipients are involved in each message, creating co-occurrence patterns that an adversary can exploit. At the same time, instant messaging imposes strict usability constraints: users expect short delays, communication is highly interactive and bursty, and activity levels fluctuate significantly over time. These properties challenge the applicability of traditional anonymity systems and raise a critical question: **Can mix networks provide strong anonymity for group messaging under realistic conditions?**

In this work, we present the first systematic analysis of anonymous communication networks in the context of instant group messaging, grounded in real-world data from WhatsApp [62]. We show that different approaches, including Signal, Tor, and mix networks, are vulnerable when used for group messaging. Specifically, we introduce a novel attack that leverages co-occurrence patterns of recipients over time to infer group membership with high accuracy. We evaluate this attack under realistic conditions and demonstrate its effectiveness even against architectures such as Loopix [51].

For this purpose, we adapt real-world data to better reflect realistic usage scenarios and assess how established defensive strategies, including diverse delay techniques and cover traffic, perform under these conditions. Our findings reveal fundamental weaknesses in current designs and provide insights into how to provide anonymity in instant group messaging.

2 Related Work

Various systems have been developed to provide anonymity in communication, ranging from low-latency onion routing networks to decentralized messaging applications. Tor [20] is the most widely used anonymity system, primarily designed for web browsing, and achieves privacy by routing traffic through multiple relays with layered encryption. Other systems, such as I2P [77], focus on peer-to-peer communication and hidden services, providing anonymity within an internal network. In addition, several anonymous messengers aim to reduce metadata exposure. For example, Signal’s Sealed Sender feature [65] hides the sender’s identity from the server, Ricochet [8] uses Tor hidden services for peer-to-peer messaging,



This work is licensed under a Creative Commons Attribution 4.0 International License. *CODASPY '26, Frankfurt am Main, Germany*
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2562-3/2026/06
<https://doi.org/10.1145/3800506.3803513>

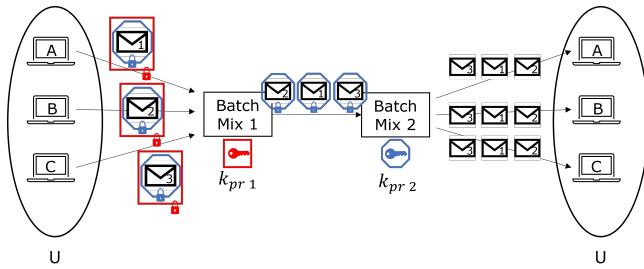


Figure 1: Functionality of a simple Batch Mix network

and Briar [3] relies on Bluetooth or Wi-Fi Direct for decentralized communication.

Mix networks, first introduced by Chaum [10], are a fundamental approach to achieving anonymous communication. Figure 1 displays this original protocol using Batch Mixes: Instead of sending their messages directly to recipients, users utilize a route consisting of multiple mix nodes. They recursively encrypt their messages with the public keys k_{pub_i} of the mixes on the route before forwarding them to the first mix, e.g., Alice encrypting her message with k_{pub_2} and then k_{pub_1} . Each mix collects the same number of messages from each user of the user set U (e.g., one message from Alice, Bob, and Charlie), decrypts one layer of encryption with their private key k_{pr_i} , and forwards them, in a shuffled order, to the next node. This process is repeated until the final mix broadcasts the messages to all users. Furthermore, the mixes prevent replay attacks by dropping repeated packets.

To improve usability, several variants have been proposed. Timed Mixes [22], also employed by MixMaster [41] in combination with a pooling approach, release messages at fixed time intervals rather than waiting for a full batch, guaranteeing specific delays. Stop&Go Mixes [32] take a different approach: each message is delayed by a random value drawn by the senders from an exponential distribution, and packets include timing constraints to prevent malicious delays. Loopix [51] builds on Stop&Go principles and introduces providers that allow clients to remain offline between transmissions. These designs relax synchronization requirements and improve practicality for asynchronous communication. In contrast, Pool Mixes [41] select messages randomly for forwarding, which can provide strong anonymity but cannot guarantee timely delivery, a critical requirement for instant messaging. For this reason, Pool Mixes are not considered in our work.

Many different anonymization systems (e.g., Dissent [73], Ripeste [12], Vuvuzela [70], or the real-world multi-purpose implementation Nym [17]) have been proposed over time, but they are not suited for instant (group) messaging due to multiple reasons. Several works assume synchronous environments and require users to be either online at all times or synchronize their sending times, which is not suitable for mobile instant messaging and global user bases. Furthermore, works often do not consider the short delay expectations of instant messaging or introduce costly overhead, and even approaches specifically mentioning instant messaging, such as Loopix [51], have usually focused on one-on-one communication.

To the best of our knowledge, we are the first work to analyze the suitability of mix networks for anonymizing instant group messaging using a real messenger data set. Previous evaluations of anonymity systems have often relied on synthetic traffic models or datasets from domains such as web browsing [24, 33], e-mail communication [16, 46, 47, 53], and social networks [43, 76]. IRC chat logs [74] and microblogging platforms like Twitter and Reddit [25] have also been explored, though these differ significantly from private instant messaging in terms of interaction patterns and latency expectations. For our analysis, we use the WhatsApp dataset published by Seufert et al. [62], which provides anonymized private chat histories and enables realistic modeling of group communication. While other datasets exist, such as public WhatsApp group data [26], they focus exclusively on public groups and are therefore unsuitable for evaluating anonymity in private messaging contexts. Additionally, other works have used real-world data from Telegram [59] and WhatsApp [52] to develop communication models, though they focus on contact networks instead of messaging behavior.

This work builds upon previous messenger mix network analysis [58], utilizing its communication turn model. However, we compare previous modeling with real messaging behavior and user expectations, analyze a generalized anonymization system, implement a concrete group deanonymization attack, and evaluate it against a realistic mix network, considering different defense strategies.

3 Modeling vs. Practice: Instant Messaging Behavior and Anonymity Networks

As discussed in the previous Chapter, some studies evaluating mix networks use real-world datasets, most commonly e-mail, but these differ significantly from instant messaging in terms of communication patterns and user behavior. Understanding these differences is essential for evaluating whether existing defenses remain effective under realistic conditions. Furthermore, most research relies on statistical models to simulate user behavior and network structure.

3.1 Assumptions in Mix Network Communication Modeling

Early works demonstrating the effectiveness of Disclosure [31] and Statistical Disclosure Attacks [13–15] often assumed uniform communication patterns among contacts. This assumption persists in later studies that extend statistical or data-driven approaches [47, 50]. Background messaging behavior is frequently modeled using normal distributions [39, 40] or zeta distributions [50], while target user activity is simulated using geometric [2], Poisson [53], or multinomial distributions [48].

The structure of the contact network is also abstracted using graph models. Several studies employ scale-free networks [6], which capture the presence of highly connected nodes and have been used in works such as [2, 19, 39, 40, 43]. Others use Watts-Strogatz small-world networks [19, 43, 57] or expander graphs [43] to simulate realistic user connections.

3.2 Real-World Instant Messaging Behavior

To assess how well the assumptions made in mix network research align with real-world usage, we reviewed the literature on instant messaging behavior. While many studies focus on qualitative aspects such as use cases and social impact (e.g., [29, 34, 42, 64, 67]), we summarize key findings from quantitative research that are relevant for modeling communication patterns.

Contact networks in instant messaging are typically small and highly clustered. For most users, the core network consists of only a few individuals: 50% of text messages go to just five contacts [37], and 76% of users add fewer than five new contacts per month [36]. In work-related contexts, 50% of users interact with fewer than 14 people daily [64]. These networks are often modeled as small-world graphs [71], with clustered structures and power-law degree distributions [52], similar to social [44] and e-mail networks [21], though other works have observed different patterns, including Weibull distributions [75].

Messaging behavior, however, differs significantly from the assumptions in many mix network models. Instant messaging is characterized by **bursty, session-based communication**: short, active exchanges followed by long idle periods. These busy phases typically include 14–26 messages [4], with only 6.9% of messages initiating new sessions [5]. During active phases (conversations with pauses shorter than 30 minutes), users respond rapidly—within one minute in 69.9% of cases [63], and five minutes in over 90% in [5]. In our dataset [62], we observe an average response time of 2 minutes and 19 seconds (when considering replies within 30 minutes). Even without distinguishing active phases, in one work 40.7% of responses occur within five minutes [54] and, in another, 57.8% within one minute [55].

These patterns contrast sharply with the **uniform or memoryless activity models** (e.g., Poisson or geometric distributions) often used in mix network simulations. The high responsiveness and temporal clustering in instant messaging pose challenges for anonymity systems, which can be even more pronounced in group chats, where multiple recipients create correlated observable events.

3.3 User Expectations

Compared to previous technologies like SMS, IM is used for its short delays, enabling "dynamic, fluid replying behavior" [11] with the expectation of immediate answers [1]. However, these expectations can vary depending on the scenario and user group. On one hand, experiencing 90-second delays from partners increases frustration [30], and messaging apps that take ten or more seconds to load images see a drop in acceptance rate of ten percentage points [61]. In other contexts, only 28% of users expect a reply in less than 5 minutes from customer support live chats [68]. These expectations can be influenced further if indicators showcase that a message has been read, causing lower communication satisfaction when combined with delays [23]. While some guidelines [69] recommend keeping delays below 10s, other studies have concluded that the maximum tolerable delay in voicemail, depending on the urgency of the situation, ranges between 15.8s and 158.4s [9]. Based on these differing requirements, we analyze how well anonymity can be improved if larger message delays are accepted in Section 8.4.

4 Data Set and Challenges

As examined in the previous chapters, prior work on mix networks often relies on simplified assumptions or datasets that do not reflect the characteristics of instant messaging. To address this gap, we utilize the dataset published by Seufert et al. [62], which comprises anonymized WhatsApp chat histories submitted by users through an online tool. The dataset was collected through voluntary contributions, where participants exported selected (group) chats from WhatsApp. The tool automatically anonymized the data by replacing message content with metadata (e.g., length) and substituting user identifiers with randomized IDs.

The analyzed dataset comprises a total of 5,956 chats, containing 76.9 million messages. Of these, 2,290 are direct chats between two individuals (32.6m messages) and 3,666 group chats (44.3m messages). These messages span multiple years, as shown in Table 1. We treat all messages the same, even though information about message types (e.g., media or text), length, and emoji usage is available, as this is not the focus of our work. A sample of the dataset used in our experiments is shown in Table 2.

While the anonymization limits the available information, the data set retains the essential elements—sender, recipient(s) (indicated by the chat), and timestamp—required for simulating anonymization techniques. However, working with this dataset presents several challenges that must be addressed to ensure a realistic and meaningful evaluation:

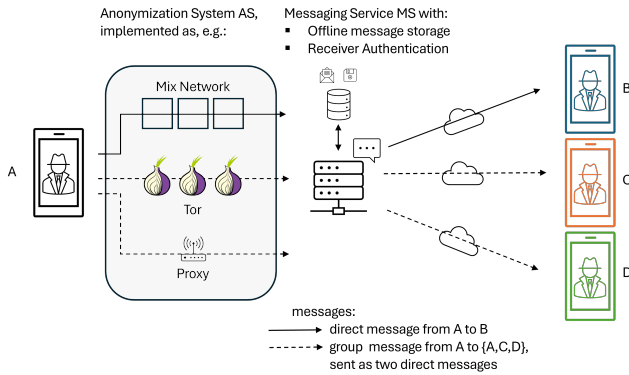
- **Dataset Size and Representativeness:** Even though the dataset spans ten years, there is significant variation in message density across them, with 85% of messages occurring in the most active four years, directly impacting reachable anonymity levels. Furthermore, it may not capture the diversity of usage patterns found in larger or more heterogeneous populations due to the self-selection of participants and their submitted chats, introducing a potential sampling bias.
- **Temporal Localization Bias:** WhatsApp exports use the local time of the exporting device. Although the dataset includes users from different countries, all timestamps appear, like in Table 2, as if they occurred in the same time zone. This distorts temporal patterns and may not reflect the asynchronous nature of global communication networks.
- **Lack of Global User Identity:** Since user identifiers are randomized per chat, it is not possible to recognize the same user across different chats. Even though the most prominent user submitted 128 chats, they cannot be reidentified. As a result, the system needs to be adapted to model broader communication patterns in social networks.
- **Message Dependency:** Messages in instant messaging are not independent events; they often form part of a conversation, with replies depending on previous messages. However, the data set does not provide insight into the relation between messages. For example, in Table 2, the fourth message occurred 4 minutes after the third one from the other user in the chat, indicating that it might be an answer, though there is no way of confirming this based on the data available. Ignoring these dependencies can lead to unrealistic assumptions about user behavior.

Table 1: Messaging statistics per year

year	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
active chats	5	35	124	269	651	1412	2864	3293	2280	1585	652
messages	3.8k	59.7k	213.7k	654.0k	1.6m	5.2m	17.9m	20.6m	16.2m	10.8m	3.5m

Table 2: Excerpt of the messages utilized

message	chat	sender	recipient	time
13715	0 (group)	1	2	2017-10-07 01:47:00
13715	0 (group)	1	3	2017-10-07 01:47:00
13716	1 (1-on-1)	4	5	2017-06-08 17:50:00
13717	1 (1-on-1)	5	4	2017-06-08 17:54:00

**Figure 2: Overview of the analyzed generic anonymization system model**

These limitations highlight the importance of adapting both simulation and evaluation methods to account for the constraints of real-world data, and we describe how we address these challenges in Chapter 8.3.

5 System Model

In this chapter, we define a generic system model for anonymous Mobile Instant Messaging (aMIM), which serves as the basis for evaluating existing anonymization approaches. We define multiple requirements that shape our target architecture:

- **Messaging:** Users must be able to send and receive messages. In particular, we want to support group chats.
- **Instant Messaging:** Messages must be delivered in a timely manner, meeting user demands (see Section 3.3). While instant messaging, due to minimal delays, can enable (near) synchronous communication if two users are online at the same time, we focus on asynchronous communication, i.e., there are delays between receiving a message and answering it.
- **Mobile Instant Messaging:** Users are not always online; the system needs to provide message storage for offline users.

Furthermore, users should be able to act independently of each other and the messaging system.

- **Anonymous Mobile Instant Messaging:** We define *relationship anonymity* as our privacy goal: This includes the established *sender-recipient unlinkability* [49] (i.e., an attacker should not be able to determine who communicates with whom) and also *undetectable groups*, meaning that an attacker outside of chat groups cannot deduce their members or detect if two users share one or more groups. Specifically, this requirement shall also protect users who do not actively participate in group chats ("silent readers"), a relevant user base due to uneven group contributions (see e.g., [62]), and the possibility to restrict who can send messages inside group chats in current instant messengers (e.g., Signal, WhatsApp, Telegram).

Building upon these requirements, we present our target architecture in Figure 2. Requirements 1 and 2 simply create the need for a fast messaging system. To support offline users, we include a Messaging Service (MS) that stores messages for recipients, who can request them at a later time by authenticating themselves. Since the MS knows the message recipients, users must be able to send anonymously to achieve *sender-recipient unlinkability*. Thus, there is an Anonymization System (AS) between senders and the MS. We do not further specify the AS or how senders interact with it. For example, as long as the messaging protocol does not require senders to authenticate/identify themselves against the MS, users could employ Tor, a proxy, or a mix network, and our attack applies regardless of the choice. This mix network can be further modified and, for example, include in-providers, as seen in Loopix [51], which also provides sender but not receiver anonymity.

We treat the AS and how senders interact with it as a black box. It is only constrained by two requirements: First, it must deliver messages fast enough to meet user expectations. Second, the group message delivery: In many commercial messaging platforms, e.g., WhatsApp, the server knows all groups and their members and is responsible for distributing group messages to all members [72]. Since this would directly violate the *undetectable groups* requirement, we instead look towards more privacy-focused messaging applications. In Signal, only users manage and know of the existence of their groups. Group messages are sent as individual direct messages to all other group members, which the Signal server cannot distinguish from regular direct messages [56]. Thus, we also assume that direct and group messages are indistinguishable for the MS. This could be achieved by the end user sending group messages as individual messages or being delegated to the AS for performance reasons for mobile users.

The assumed system model we analyze is flexible, as we treat the sending process as a black box, and the MS can be represented by one or more servers that users choose from. As a result, the

evaluation results presented in this work remain applicable across many different scenarios. Now, it might seem trivial to also include anonymization on the receiver side. However, such systems will have a hard time fulfilling the requirements of MIM, as, e.g., the Anonymous Post-office Protocol [27] requires clients not to be offline for too long.

6 Group Identification Attack

6.1 Basic Idea

Based on the generalized system architecture of the previous chapter, we now present our attacker model and the group identification attack. Our attacker is a passive local attacker, controlling the MS (or a fraction of them), i.e., honest but curious. When a message arrives at the MS, the attacker learns the arrival time and the message recipient, but he does not know the sender or whether the message is a direct message or a group message¹. Now, while this attacker is very limited in a classical scenario consisting only of one-on-one communication, when group chats are involved, he can observe group members appearing closely together on the receiver side. Figure 3 demonstrates the occurring patterns. Because group messages are sent to each group member (other than the sender), group members will receive messages at similar times and in comparable amounts (though active users will receive fewer messages than passive ones since they themselves sent more). In our proposed attack, the attacker now executes three steps to identify groups:

- (1) Closeness calculation: Based on users' message receiving times, the attacker assigns an asymmetrical closeness score to each user pair.
- (2) Relationship graph construction: The attacker creates a directed weighted graph where recipients are represented by vertices and the edges between them are weighted by the closeness scores determined in step one.
- (3) Determination of groups: The attacker identifies groups (or contacts) based on their closeness and the connectedness of all group members.

In the first step, to calculate the closeness of users, the attacker uses the arrival times of messages. We showcase this calculation based on the example shown in Figure 4. When the first message for user A_1 arrives at t_1 , the attacker stores that message with its recipient, time, and an empty closeness map. When the next message (for user A_2) arrives at t_2 , its closeness to all stored messages with a recipient other than A_2 is calculated. The closeness score is calculated as in Equation 1.

$$cl_{msg}(t_1, t_2) = \exp\left(-\frac{|t_1 - t_2|}{E(d)}\right) \quad (1)$$

$E(d)$ represents the expected delay of a message sent through the AS and helps shape the score curve. This results in a message closeness score between 0 (messages are very far apart) and 1 (messages arrived at the same time). This score is then added to both the closeness maps of messages one and two. (For message one, the map now contains the entry A_2 with the calculated value.) When

¹Additionally, the MS can also see when users (receivers) request their messages. The attacker could use this information to determine if later messages originate from certain actors, based on their online appearances, though we leave this out for future work.

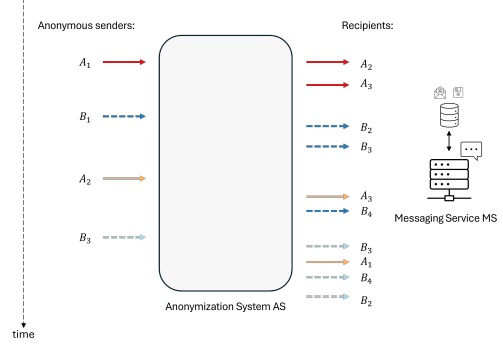


Figure 3: Group message recipient patterns: Group members (except the sender of a message) receive group messages at similar times.

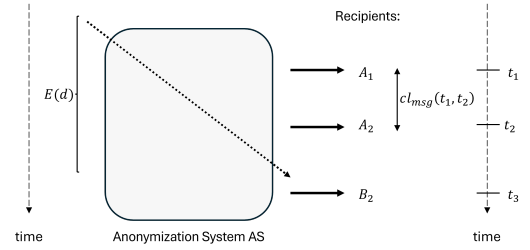


Figure 4: Message closeness calculation

Table 3: Overview of all message closeness scores^{*} resulting from the example in Figure 4

other recipient	message closeness score
message 1 (recipient: A_1 , arrival time: t_1)	
A_2	$cl_{msg}(t_1, t_2)$
B_2	$cl_{msg}(t_1, t_3)$
message 2 (recipient: A_2 , arrival time: t_2)	
A_1	$cl_{msg}(t_1, t_2)$
B_2	$cl_{msg}(t_2, t_3)$
message 3 (recipient: B_2 , arrival time: t_3)	
A_1	$cl_{msg}(t_1, t_3)$
A_2	$cl_{msg}(t_2, t_3)$

^{*} assuming all messages are close enough to each other to not fall below the closeness threshold ϵ .

the next message arrives, its closeness to all other stored messages with a different recipient is calculated and stored symmetrically. Table 3 shows the calculated closeness maps for each message after all messages from Figure 4 are processed.

To not store messages indefinitely, once the closeness between a stored message and a newly received message falls below a threshold ϵ (e.g., 0.01), the stored message is removed from the message list. To permanently store the message closeness values cl_{msg} from

this message, first, for normalization, all values greater than one are reduced to one, and then added to the recipient's overall user closeness map cl_{user} . Furthermore, the total number of messages received rec_u by the user u is increased by one. At the end of the observed time frame, the remaining stored messages are also processed in the same way, and the final user closeness scores for each user pair can be calculated as shown in Equation 2. This represents the edge from user u_1 to user u_2 , and the edge going in the opposite direction is calculated as $cl_{edge}(u_2, u_1)$.

$$cl_{edge}(u_1, u_2) = \frac{cl_{user}(u_1, u_2)}{rec_{u1}} \quad (2)$$

After the data collection, the attacker has now constructed a weighted directed graph, where nodes represent users (recipients) and for each user pair (u_1, u_2) that received messages close enough to each other, there are two connecting edges, weighted differently. An example of such a graph is given in Figure 5 based on the message arrivals from Table 4, with the groups $A = \{A_1, A_2, A_3\}$ and $B = \{B_1, B_2, B_3\}$. Worth noting is the missing connection between users A_3 and B_2 because their closest appearances were too far apart, falling below the threshold ϵ . Furthermore, since A_1 sends most messages in group A, he appears less often as a recipient and thus his closeness scores $cl_{edge}(A_1, A_i)$ are greater than the ones going in the opposite direction. Because B_1 only sent but never received messages, he does not appear in the attacker's dataset or graph.

6.2 Single Group Attack

We first describe the group identification attack for a simplified scenario: All users appear only in one group. While this does not correspond to reality, it nevertheless describes the initial situation of the data set due to the lack of global user identities. We explain how we later adapt the data to reflect real IM behavior with users participating in multiple chats in Section 8.3, and show how we adapt the attack in the next section. Assuming that each user is only part of one group, the problem can directly be expressed as community detection in a graph (see, e.g., [35] for an overview), i.e., determining user clusters (chat groups) based on their relationship (edge weights). The attack algorithm now follows three steps to determine groups:

- (1) For each user u_i , calculate a closeness threshold $cl_{threshold}(u_i)$ (i.e., how close do other users need to be to them to be considered a group member) based on the maximum of all their outgoing edges $cl_{edge}(u_i, u_j)$ and a variable rate α (between 0 and 1) as $cl_{threshold}(u_i) = \alpha \max_{u_j \in U} (cl_{edge}(u_i, u_j))$.
- (2) Construct a new undirected unweighted graph with the same user nodes but starting without connecting edges. For each user pair (u_1, u_2) , calculate the mean of their connection scores $cl_{mean}(u_1, u_2) = \frac{1}{2}(cl_{edge}(u_1, u_2) + cl_{edge}(u_2, u_1))$. If this mean score is at least as big as either user's closeness threshold $cl_{threshold}(u_i)$ from step one, construct the undirected edge between u_1 and u_2 . Based on the previous example closeness graph from Figure 5 and $\alpha = 0.8$, the resulting relationship graph is depicted in Figure 6.
- (3) Now, we perform a (depth-first) search of the resulting graph to identify groups (connected users), only saving groups of at least size 3 and with a minimum internal density of 0.5.

Table 4: Sample message arrivals from the groups $A = \{A_1, A_2, A_3\}$ and $B = \{B_1, B_2, B_3\}$

msg id	1.1	1.2	2.1	2.2	3.1	4.1	4.2	3.2	5.1	5.2
sender	A_1	A_1	A_1	A_1	B_1	A_3	A_3	B_1	B_1	B_1
recipient	A_2	A_3	A_3	A_2	B_3	A_1	A_2	B_2	B_3	B_2
arrival time	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}

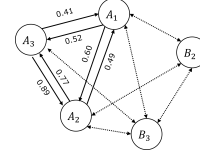


Figure 5: Closeness graph resulting from the message arrivals from Table 4 with example weights. Dotted connections do exist, but edge weights are omitted for visual clarity

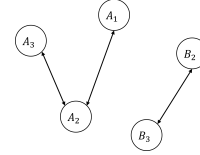


Figure 6: Example relationship graph resulting from the closeness graph from Figure 5 with $\alpha = 0.8$. This would result in the groups $\{A_1, A_2, A_3\}$ and $\{B_1, B_2, B_3\}$.

Basing the closeness threshold from step one on each user's connections allows the algorithm to work for active and passive group members. E.g., in Figure 5, user A_1 has lower incoming than outgoing closeness values to his group members due to being a more active sender (and thus appearing less often as a close recipient to his group members). However, the choice of α is not easy. Therefore, the algorithm is executed multiple times for different values of α (e.g., from 0.2 to 0.8 with increments of 0.05). The algorithm then has to estimate which of the resulting candidate graphs and groups represent reality best. To do so, it rates the resulting groups based on the mean of two normalized metrics and chooses the best result:

- Coverage: The fraction of users assigned to a group, i.e., with at least one connection in the undirected relationship graph.
- Average group density: For each group, compute the group's density, i.e., the ratio of edges compared to the maximum possible number of undirected edges inside the group as $\frac{|E|}{\frac{1}{2}|V||V|-1}$, where $|E|$ is the number of edges and $|V|$ the number of vertices. Then, compute the (unweighted) mean density of all groups.

The graph from Figure 6 would result in the groups $\{A_1, A_2, A_3\}$ and $\{B_1, B_2, B_3\}$ with a coverage of $\frac{5}{5} = 1$ and density $\frac{1}{2}(\frac{2}{3} + \frac{1}{1}) = 0.83$, yielding an overall score of 0.92.

6.3 Multiple Groups per User

Now, this algorithm will fail if applied to a real scenario where users are part of multiple groups. The current implementation expects a

relationship graph with disjoint clusters (one per group). However, if Alice is part of groups A and B, she might be a connecting bridge between the two groups, resulting in merged groups. Furthermore, each user's closeness threshold $cl_{threshold}(u_i)$ is calculated based on the assumption that all their connections are to the same group. If group A is more active than group B, the algorithm might base Alice's threshold on group A, resulting in Alice's connections to group B being too small to notice. Thus, we adapt the algorithm into an iterative peeling process to allow for multiple groups per user:

- (1) detect the strongest, densest groups on the current directed closeness graph;
- (2) subtract the mean of their intra-group edge weights;
- (3) re-run detection on the residual graph to reveal weaker or overlapping groups;
- (4) repeat until no meaningful new groups appear.

We keep the original mechanics for calculating message closeness from close recipient appearances. We then still identify groups by running the group-detection algorithm for multiple values of α . After accepting a group, we reduce its intra-group directed edges by the mean of those edges after this iteration. This “deflates” the already-explained structure, allowing weaker, previously hidden groups to surface. To avoid (near-)duplicates, every new candidate group is compared against the previously accepted groups: If it is an exact match or highly similar (Jaccard index (Equation 3) $J \geq 0.8$ or one group contains at least 80% of the other one), it is skipped.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \quad (3)$$

Furthermore, in the multi-group setting, candidate components in the acceptance graph often contain “bridge” users who share several groups and thereby reduce internal density. To better recover specific groups without changing the overall attack structure, we add a substructure extraction step that is applied only when a connected component does not meet the minimum density requirement. Starting from the full component, it repeatedly removes the currently least connected member with respect to edges inside the component, while updating the remaining internal connectivity. During this process, we track the best intermediate subgraph encountered and stop early once the subgraph reaches a high density level. In our experiments, this early-stopping criterion works best with a target density of 80%. Importantly, this 80% target is used only as an internal stopping and selection heuristic, and it does not replace the global acceptance threshold. After this process finishes, the resulting subgraph is still filtered using the regular criteria, and it is still subject to our duplicate and near-duplicate suppression. This modification improves recall in the multi-group scenario by extracting a dense core from otherwise mixed or loosely connected components, though this does reduce the result's precision.

We stop when the residual graph no longer yields groups. Peeling strong groups first and re-detecting on the residual reveals overlapping memberships and weaker communities that the original single-run detector would miss due to the dominance of highly active groups. Figure 7 showcases the adapted algorithm for a simple scenario with two overlapping groups. In the first step (left), the more active group $\{A, B, C\}$ is detected, and after its mean edge

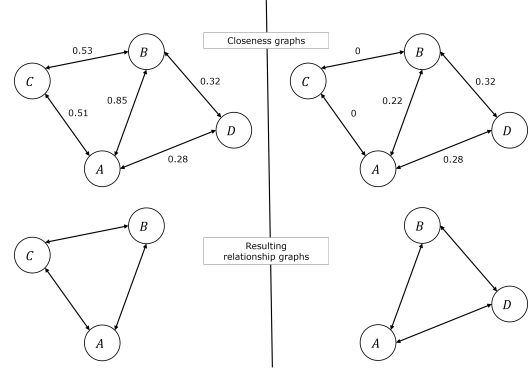


Figure 7: Example run of the multi-group algorithm for a scenario with the groups $\{A, B, C\}$ and $\{A, B, D\}$. (Showing mean edge weights, and the low value edge CD is omitted.)

weight 0.63 is subtracted, the remaining group $\{A, B, D\}$ can be detected in step two (right).

7 Anonymity Metrics

To evaluate the effectiveness of our proposed group identification attack, we require metrics that accurately capture its success. At the same time, it is essential to demonstrate that the scenarios under consideration are not trivially deanonymizable using generic techniques.

Therefore, this chapter is structured as follows: we first present classical anonymity metrics that are widely used to assess the anonymity properties of communication systems. These metrics allow us to establish the baseline anonymity level in our analyzed scenarios and confirm that they are non-trivial to attack. Subsequently, we introduce our tailored evaluation metrics, specifically designed to measure the performance of our group identification attack. These specialized metrics focus on the accuracy and completeness of group reconstruction, providing a clear view of how well our attack compromises anonymity in the context of instant group messaging.

7.1 Classical Anonymity Metrics

A variety of metrics have been proposed to quantify anonymity in communication systems. One of the earliest and most intuitive is the anonymity set size [49], which measures the number of participating users who cannot be distinguished by an attacker. Some works refine this by considering the number of active users at a given time [7], which better reflects the anonymity available in dynamic systems.

To capture more nuanced anonymity levels, especially in probabilistic or non-uniform settings, entropy-based metrics have been introduced [18, 60]. These consider the probability distribution over possible senders or recipients, with higher entropy indicating greater uncertainty for an adversary.

In addition to established measures, we incorporate concepts originating from database anonymity, namely k -anonymity [66] and l -diversity [38]. These concepts provide a structured way to reason about anonymity systems like mix networks. In our context,

Table 5: Example values of entropy for different batches

senders	entropy	max entropy	normalized entropy
1,2	1	1	1
1,2,3,3,3,3,3,3	0.99	1.59	0.62
1,1,2,2,3,3,4,4	2	2	1
1,2,2,3,3,3,4,4,4,4	1.85	2	0.93

k-anonymity corresponds to the batch size: the number of messages processed together determines the size of the anonymity set. However, instant messaging traffic is highly bursty, and group messages are implemented as multiple direct messages. As a result, a batch may be dominated by a single sender, reducing effective anonymity. To capture diversity, we therefore also consider l-diversity, defined as the number of unique senders or recipients within a batch.

The composition of each batch can be further analyzed utilizing entropy l-diversity [45]: the Shannon entropy of the sender (recipient) distribution within a batch, as in (4), where p_s is the proportion of messages from sender s in batch S . However, raw entropy can be misleading. For example, as demonstrated in Table 5, two batches may have similar entropy values even if one is perfectly balanced and the other is dominated by a single sender. To account for this, we use normalized entropy, defined in (5), representing the fraction of the maximum achievable entropy reached, where l is the number of distinct senders in the batch. This yields a value between 0 (completely skewed) and 1 (perfectly uniform), providing a clearer picture of batch homogeneity. We define the (normalized) entropy of a batch with only one sender as 0, reflecting the absence of anonymity.

$$H(S) = - \sum_{s \in S} p_s \log_2(p_s) \quad (4)$$

$$H_n(S) = H(S) / \log_2(l) \quad (5)$$

Now, these metrics were designed for systems with distinct batches (e.g., Batch, Timed, or Pool Mixes). However, we want to analyze a generalized AS that might operate continuously (e.g., a mix network utilizing Stop&Go Mixes [32]). Thus, we instead measure the amount of other messages/recipients/groups present when messages enter and leave the AS.

Finally, when evaluating anonymity, it is not sufficient to report average values. A system may appear secure on average while still exposing users in a subset of cases. Thus, to achieve k-anonymity and l-diversity, all equivalence classes must fulfill the condition k or l . To capture this, we analyze the **distribution** of anonymity values across all messages and report lower quantiles, e.g., the 10% quantile (denoted as 10% Q) to reflect the minimum anonymity level achieved by 90% of messages.

7.2 Group Identification Attack Metrics

In addition to the classical metrics introduced above, we now present two specialized measures to evaluate the success of our group identification attack. Each metric captures a complementary perspective: *pairwise* correctness of co-membership, and *cluster-level* fidelity of whole-group boundaries.

Pairwise Co-Membership (Precision / Recall / F1): This metric evaluates correctness at the level of *user pairs*. Let $\mathcal{P}^*$ be the set of ground-truth pairs that co-occur in at least one real group and $\mathcal{P}$ the set of predicted user pairs (users sharing at least one predicted group). We compute Precision $pair_p = \frac{|\mathcal{P} \cap \mathcal{P}^*|}{|\mathcal{P}|}$, Recall $pair_r = \frac{|\mathcal{P} \cap \mathcal{P}^*|}{|\mathcal{P}^*|}$, and F1 $pair_{F1}$ as the harmonic mean. Intuitively, high Precision indicates few false merges (pairs predicted together but not truly co-members), while high Recall indicates few misses (true co-member pairs recovered by the prediction).

Cluster-Level One-to-One Matching (Size-Weighted Jaccard): This metric assesses the fidelity of *whole group boundaries*. We align predicted groups G_j to ground-truth groups T_i via a maximum-weight bipartite matching using Jaccard overlap as the edge weight. We then compute a *size-weighted mean Jaccard* over matched pairs, weighting by the size of ground-truth and predicted groups.

As an example, assume the true groups $T_1 = \{A, B, C\}$, $T_2 = \{D, E\}$ and the predicted groups $G_1 = \{A, B, C\}$ (perfect prediction of T_1), $G_2 = \{D, E, F\}$ (covers T_2 but falsely includes F). Now, the algorithm computes the Jaccard index for all possible pairings (T_i, G_j) and selects the highest score (best match) first: $J(T_1, G_1) = 1.0$. All matches containing at least one of the groups are discarded, and the next best available match is selected: $J(T_2, G_2) = 0.667$. All groups are matched, so now the algorithm scores all matches. First, weighted by the size of the real groups T_i , resulting in a recall-like score: $cluster_r = \frac{3 \cdot 1.0 + 2 \cdot 0.667}{3+2} = 0.867$. Then, weighted by the size of predicted groups G_j (precision-like score): $cluster_p = \frac{3 \cdot 1.0 + 3 \cdot 0.667}{3+3} = 0.833$. Resulting in a harmonic mean $cluster_{F1} = 0.85$.

If unmatched groups remain, they impact the denominator of either the recall component (true groups unmatched) or the precision component (predicted groups unmatched). High scores indicate that predicted groups closely match true groups without excessive merges or splits. The score penalizes fragmentation (one true group covered by several predicted groups) and over-aggregation (one predicted group spanning multiple true groups). We further also report the number of perfectly identified groups.

8 Evaluation

8.1 Implementation

This chapter investigates the vulnerability of anonymized instant group messaging to our group identification attack under realistic conditions, using the dataset introduced in Chapter 4. We simulate all messages, creating a realistic messaging network pattern, but only evaluate the algorithm's performance on detecting group chats (i.e., chats with at least three participants). In our simulation, all messages are processed in chronological order based on their original timestamps. Messages enter the AS immediately and are released after being delayed. For our AS, as a baseline, we choose a mix network consisting of $k = 3$ Stop&Go Mixes with delay rate λ (mean delay per mix $1/\lambda$). This sequence of exponential delays results in an overall delay following an Erlang Distribution (shape k , rate λ) with expected mean delay $E(d) = k/\lambda$. To enable fast messaging while still protecting anonymity, we choose $\lambda = 1/\text{min}$, resulting in a mean delay of three minutes. We analyze different delay durations and strategies in Section 8.4.

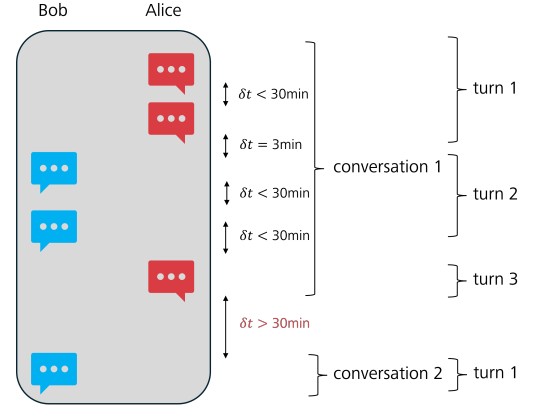
Table 6: Evaluation results of the single group attack on the original data

time frame	2015-01		2016-01	
	1 week	month	1 week	month
chat stats				
1-1 chats	55	73	161	202
group chats	99	127	274	323
mean msg per group chat	103	367	110	455
anonymity metrics				
msg: mean	192	184	411	429
msg: 10%Q	20	20	84	106
groups: mean	10	10	23	24
groups: 10% Q	4	4	9	11
groups H_n : mean	0.71	0.73	0.76	0.76
groups H_n : 10%Q	0.49	0.52	0.63	0.63
receivers: mean	72	68	174	177
receivers: 10%Q	12	12	46	58
attack results				
predicted groups	71	100	93	244
perfect groups (% of real)	41(41%)	88(69%)	43(16%)	142(44%)
$cluster_{F1}$	0.70	0.85	0.37	0.66
$cluster_p$	0.92	0.96	0.91	0.92
$cluster_r$	0.56	0.77	0.23	0.52
$pair_{F1}$	0.76	0.87	0.28	0.65
$pair_p$	0.94	0.88	0.98	0.95
$pair_r$	0.64	0.86	0.17	0.50

8.2 Evaluation of Raw Data

To establish default results, we first perform our analysis on the original raw data without modification (except for modeling group messages as multiple direct messages delayed independently). For each message, we record the delay experienced, as well as the number of other messages and groups present, and their distribution. We omit network and processing delays, as these are typically negligible compared to the mix-induced delay and vary depending on the underlying infrastructure.

Table 6 presents the results of the initial analysis for January 2015 and 2016, illustrating the attack’s effectiveness when observing messages for one week or one month. In 2015, lower overall chat activity led to higher success: after one week, 41 of 99 group chats were perfectly identified, and even imperfect predictions showed high precision, with 94% of connected user pairs actually sharing one or more chats. With a full month of observations, 69% of group chats were identified perfectly. In 2016, higher activity and thus more anonymity reduced the number of perfectly identified groups, although precision remained very high and a substantial number of chats and connected users were still identified, with clear improvements when observing a full month rather than one week. Finally, the algorithm shows no clear bias toward small or large groups, as both the average size of all real chats and of perfectly identified chats is 24 in the 2016 full-month analysis.

**Figure 8: Concept of conversation turns**

8.3 Simulation Adaption

However, this first analysis remains highly idealized. It assumes that messages are independent and can be sent at any time, ignoring conversational dependencies (i.e., replies). These temporal interconnections constrain when messages can be sent, which in turn affects the achievable anonymity. Since the dataset does not indicate which messages are replies, we approximate this behavior by modeling **conversation turns**, using the approach from [58]. We define a conversation turn as a change of sender within an active session (conversation). We assume that if two consecutive messages in a chat are sent within 30 minutes of each other ($\delta t < 30min$) and the sender of the second message differs from the first, the second message is a response. This heuristic allows us to infer a basic conversational structure from the message stream.

Figure 8 illustrates this concept. Applying this model to the dataset results in 6 million inferred conversations. Within these, the conversation turns have a mean length of 10 messages and a median of 4. In our simulation, we preserve the timing of these inferred responses. For example, if Bob originally replies to Alice three minutes after receiving her message, the simulation enforces this delay: Bob must first receive Alice’s message (i.e., it must be flushed from the AS), and only then will his answer be delayed by the observed $\delta t = 3min$. This can lead to long conversations being pushed outside of the analyzed time frame, resulting in some messages being sent and received when no new original messages were sent. To account for this temporal spillover, we introduce a strict cutoff at the end of the analyzed time frame. This adjustment affects only a small fraction of messages and has a negligible impact on average anonymity metrics. However, it significantly improves the accuracy of lower quantile estimates and prevents misleading conclusions about worst-case anonymity and trivially easy de-anonymization attacks.

Furthermore, each user only appears in one chat, enabling the simple Single Group Attack from Section 6.2. However, in real-world scenarios, where users are part of multiple (overlapping) groups, identifying these groups is more challenging. Thus, we create an artificial userbase with users spread across multiple chats, resulting in us applying the Multi Group Attack from Section 6.3. For our population generation, we use two input parameters: The

list of real chats (in particular, their sizes s_i) and the target mean number of chats per user $uc = 5$, based on the observations from Section 3.2. The algorithm performs two steps to assign users to chats:

- (1) Generate $N = \frac{\sum s_i}{uc}$ users, though at least as many as there are chats, with a random target degree following a power law distribution with mean uc and random 2D-coordinates.
- (2) Then, for each chat, starting with the biggest one, assign random users to its participants. To choose a user, we perform a weighted random selection, where the mean of two factors determines each user's weight:
 - Their remaining target degree (i.e., the user's original target degree minus the number of chats they are already assigned to).
 - Their normalized mean distance (based on their coordinates) to the users already assigned to the chat. This leads to local clusters and overlapping social circles instead of completely random chat groups.

We evaluate this adapted simulation environment in the next sections, analyzing the impact of different defense approaches.

8.4 Different Delay Options

While different delay strategies could be chosen to be performed by the user (e.g., different sending times) or the AS (e.g., varying delay distributions), we still apply the restrictions regarding instant messaging, i.e., there is a maximum delay between the sending and receipt of a message. To analyze the impact of varying delay strategies, this section examines the effect of selecting different delay options, so we can assess the trade-off between anonymity and latency. We keep the previous configuration with a mean delay $E(d)$ of 3 minutes, but also evaluate a faster, "more instant" messaging scenario with 30 seconds and a more secure anonymity-focused scenario with an average 10-minute delay.

The results are displayed in Table 7. To facilitate a comparison with the previous evaluation, we repeat the analysis for the first month of 2016. If the delay strategy is kept the same, the average anonymity metrics still change due to different user behavior (waiting for a message before sending answers) and identities (users appearing in multiple chats). The attack performs a lot worse when users are not only part of one distinct group. Due to iteratively reducing candidate groups until a target density is met, the algorithm no longer identifies groups perfectly and is stopped after 50 iterations. However, it is still able to recognize connected user pairs with high precision when a low delay is utilized. Furthermore, it can be seen that increasing the delay does provide protection, though with apparently diminishing returns.

8.5 Cover Traffic

However, the question remains: how can small user bases that expect short delays protect themselves? For this purpose, we analyze the efficiency of cover traffic for protecting group chats in the low-traffic scenario of week one in 2015. Since our attacker only utilizes messages arriving at receivers, anything-to-end cover traffic is required. We implement self-sent cover traffic (comparable to loopback cover traffic in Loopix [51]), which is sent by all users, independently of their activity or other users, at an exponential

Table 7: Evaluation results of the multi group attack on the adapted simulation environment

$E(d)$	2016-01		
	30s	3min	10min
1-1 chats		202	
group chats		323	
mean msg per group chat		455	
anonymity metrics			
msg: mean	65	238	554
msg: 10%Q	12	101	286
groups: mean	5	19	44
groups: 10% Q	2	12	23
groups H_n : mean	0.65	0.73	0.76
groups H_n : 10%Q	0.33	0.6	0.68
receivers: mean	54	174	365
receivers: 10%Q	10	78	205
attack results			
predicted groups	50	50	50
perfect groups	0	0	0
$cluster_{F1}$	0.33	0.11	0.07
$cluster_p$	0.63	0.23	0.17
$cluster_r$	0.23	0.07	0.04
$pair_{F1}$	0.59	0.36	0.34
$pair_p$	0.84	0.30	0.25
$pair_r$	0.46	0.45	0.51

rate. We evaluate different rates of cover traffic, and Table 8 shows the results depending on different mean delays between cover messages per user. When no cover traffic is used, the attack can already identify user pairs with high precision after one week of observations. When using cover traffic, we only track the anonymity statistics of regular messages, as there are a lot of time windows when only cover messages are inside the AS due to the low traffic scenario, leading to misleadingly low statistics, even though regular messages are protected better. The introduction of cover traffic does provide some protection from the attack, mostly introducing false positives into deduced user pairs and worsening the detection of group structures. However, even high rates of cover traffic cannot provide the same level of protection as offered by a larger user base or longer delays. This can be attributed to the constant rate of cover traffic resulting in even closeness profiles that can be calculated out by the attack, similar to how the Statistical Disclosure Attack can remove background noise [2].

9 Conclusion

This work shows that sender-anonymized group messaging creates a structural metadata leak that existing anonymity systems often overlook. Using real-world WhatsApp messages in a realistic mix-network environment and a local passive observer, we find that the attack can recover substantial group structure with high precision after short observation windows.

Increased delays improve anonymity and reduce the attack's effectiveness, but with diminishing returns. End-to-end cover traffic

Table 8: Effect of different cover traffic rates on the attack

mean cover interval	2015 - weeeek 1		
	no cover	5min	1min
1-1 chats		55	
group chats		99	
mean msg per group chat		103	
anonymity metrics			
msg: mean	37	43	45
msg: 10%Q	2	3	4
groups: mean	2.6	3.7	5.1
groups: 10% Q	1	2	2
groups H_n : mean	0.5	0.58	0.61
groups H_n : 10%Q	0	0.1	0.2
receivers: mean	29	31	33
receivers: 10%Q	1	3	4
attack results			
predicted groups	50	50	50
perfect groups	0	0	0
$cluster_{F1}$	0.39	0.32	0.22
$cluster_p$	0.46	0.41	0.28
$cluster_r$	0.34	0.26	0.19
$pair_{F1}$	0.68	0.64	0.59
$pair_p$	0.82	0.71	0.54
$pair_r$	0.59	0.59	0.64

provides protection, yet even high rates do not match the protection of larger user bases or longer delays, and can partially be averaged out by the attacker. Overall, our results suggest that short delay expectations and bursty conversations significantly limit achievable anonymity for groups, even against a weaker, local observer.

Our evaluation relies on an anonymized WhatsApp dataset, which is somewhat dated, may be subject to sampling bias, and does not provide global user identities across chats. This required simulation adaptations, including a conversation-turn heuristic and an artificial population generator to model multi-chat participation. These choices improve realism over independent messages and chats but remain only approximations.

Directions for future work include improving group reconstruction in the multi-group case and a second-stage inference: once groups are established, receiver appearances may enable attribution of messages to likely senders. On the defense, promising avenues include adaptive cover traffic that mimics conversational bursts and protocol changes that reduce deterministic co-occurrences for group delivery while remaining usable for instant messaging.

References

- [1] Annie Dayani Ahad and Syamimi Md Ariff Lim. 2014. Convenience or nuisance?: The 'WhatsApp'dilemma. *Procedia-Social and Behavioral Sciences* 155 (2014), 189–196.
- [2] Mahdi N Al-Ameen, Charles Gatz, and Matthew Wright. 2011. Sda-2h: Understanding the value of background cover against statistical disclosure. In *14th International Conference on Computer and Information Technology (ICIT 2011)*. IEEE, 196–201.
- [3] Nico Alt. 2023. Briar: Secure Messaging for Citizens and Activists during Internet Shutdowns. In *Science Peace Security '23 - Proceedings of the Interdisciplinary Conference on Technical Peace and Security Research (SPS '23)* (Darmstadt, Germany), 44.
- [4] Daniel Avrahami and Scott E Hudson. 2006. Communication characteristics of instant messaging: effects and predictions of interpersonal relationships. In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*. 505–514.
- [5] Daniel Avrahami and Scott E Hudson. 2006. Responsiveness in instant messaging: predictive models supporting inter-personal communication. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*. 731–740.
- [6] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science* 286, 5439 (1999), 509–512.
- [7] Oliver Berthold, Hannes Federrath, and Stefan Köpsell. 2001. Web MIXes: A system for anonymous and unobservable Internet access. In *Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability Berkeley, CA, USA, July 25–26, 2000 Proceedings*. Springer, 115–129.
- [8] John Brooks. 2014. *Technical design of Ricochet*. Retrieved December 3, 2025 from <https://github.com/ricochet-im/ricochet/blob/master/doc/design.md>
- [9] Barrett S Caldwell, Antoinette Derjani Bayeh, and Piyusha V Paradkar. 1996. User-Defined Control Limits for Acceptable Message Transmission Delays in Computer-Mediated Communications. In *Human Interaction with Complex Systems: Conceptual Principles and Design Practice*. Springer, 257–263.
- [10] David L Chaum. 1981. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM* 24, 2 (1981), 84–90.
- [11] Karen Church and Rodrigo De Oliveira. 2013. What's up with WhatsApp? Comparing mobile instant messaging behaviors with traditional SMS. In *Proceedings of the 15th international conference on Human-computer interaction with mobile devices and services*. 352–361.
- [12] Henry Corrigan-Gibbs, Dan Boneh, and David Mazières. 2015. Riposte: An anonymous messaging system handling millions of users. In *2015 IEEE Symposium on Security and Privacy*. IEEE, 321–338.
- [13] George Danezis. 2003. Statistical disclosure attacks: Traffic confirmation in open environments. In *Security and Privacy in the Age of Uncertainty: IFIP TC11 18th International Conference on Information Security (SEC2003) May 26–28, 2003, Athens, Greece 18*. Springer, 421–426.
- [14] George Danezis, Claudia Diaz, and Carmela Troncoso. 2007. Two-sided statistical disclosure attack. In *Privacy Enhancing Technologies: 7th International Symposium, PET 2007 Ottawa, Canada, June 20–22, 2007 Revised Selected Papers 7*. Springer, 30–44.
- [15] George Danezis and Andrei Serjantov. 2004. Statistical disclosure or intersection attacks on anonymity systems. In *International Workshop on Information Hiding*. Springer, 293–308.
- [16] George Danezis and Carmela Troncoso. 2013. You cannot hide for long: De-anonymization of real-world dynamic behaviour. In *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*. 49–60.
- [17] Claudia Diaz, Harry Halpin, and Aggelos Kiayias. 2021. The nym network. *The Next Generation of Privacy Infrastructure* (2021).
- [18] Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. 2002. Towards measuring anonymity. In *International Workshop on Privacy Enhancing Technologies*. Springer, 54–68.
- [19] Claudia Diaz, Carmela Troncoso, and Andrei Serjantov. 2008. On the impact of social network profiling on anonymity. In *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 44–62.
- [20] Roger Dingledine, Nick Mathewson, and Paul Syverson. 2004. Tor: the second-generation onion router. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13* (San Diego, CA) (SSYM'04). USENIX Association, USA, 21.
- [21] Holger Ebel, Lutz-Ingo Mielsch, and Stefan Bornholdt. 2002. Scale-free topology of e-mail networks. *Physical review E* 66, 3 (2002), 035103.
- [22] J. C. Egner. 1997. Vergleichende Analyse von Verfahren zur anonymen Kommunikation. *Diplomarbeit am Lehrstuhl für Informatik IV der RWTH Aachen* (1997).
- [23] Eulàlia Ferreres Fernández and H Engelbrecht. 2021. The Impact of Availability Expectations on Communication Satisfaction in Conversations via Instant Messaging Apps. (2021).
- [24] Volker Fusenig, Eugen Staab, Ulrich Sorger, and Thomas Engel. 2009. Slotted packet counting attacks on anonymity protocols. In *The proceedings of the Australasian Information Security Conference*. Australian Computer Society, 53–59.
- [25] Sarah Gaballah, Thanh Hoang Long Nguyen, Lamya Abdullah, Ephraim Zimmer, and Max Mühlhäuser. 2023. Mitigating Intersection Attacks in Anonymous Microblogging. In *Proceedings of the 18th International Conference on Availability, Reliability and Security*. 1–11.
- [26] Kiran Garimella and Gareth Tyson. 2018. Whatapp doc? a first look at whatsapp public group data. In *Proceedings of the international AAAI conference on web and social media*, Vol. 12.
- [27] Nathanel Gelernter, Amir Herzberg, and Hemi Leibowitz. 2017. Two cents for strong anonymity: The anonymous post-office protocol. In *International Conference on Cryptology and Network Security*. Springer, 390–412.

- [28] Andy Greenberg. 2025. *A Simple WhatsApp Security Flaw Exposed 3.5 Billion Phone Numbers*. Retrieved December 3, 2025 from <https://www.wired.com/story/a-simple-whatsapp-security-flaw-exposed-billions-phone-numbers/>
- [29] Rebecca E Grinter and Leysia Palen. 2002. Instant messaging in teen life. In *Proceedings of the 2002 ACM conference on Computer supported cooperative work*. 21–30.
- [30] Sun Young Hwang, Negar Khojasteh, and Susan R Fussell. 2019. When delayed in a hurry: Interpretations of response delays in time-sensitive instant messaging. *Proceedings of the ACM on Human-Computer Interaction* 3, GROUP (2019), 1–20.
- [31] Dogan Kedogan, Dakshi Agrawal, and Stefan Penz. 2003. Limits of anonymity in open environments. In *Information Hiding: 5th International Workshop, IH 2002 Noordwijkerhout, The Netherlands, October 7-9, 2002 Revised Papers* 5. Springer, 53–69.
- [32] Dogan Kesdoğan, Jan Egner, and Roland Büschkes. 1998. Stop-and-go-mixes providing probabilistic anonymity in an open system. In *International Workshop on Information Hiding*. Springer, 83–98.
- [33] Dogan Kesdoğan, Lexi Pimenidis, and Tobias Kölsch. 2006. Intersection Attacks on Web-Mixes: Bringing the Theory into Praxis. In *Quality of Protection: Security Measurements and Metrics*. Springer, 159–171.
- [34] Sharon Lauricella and Robin Kay. 2013. Exploring the use of text and instant messaging in higher education classrooms. *Research in Learning Technology* 21 (2013).
- [35] Jiakang Li, Songning Lai, Zhihao Shuai, Yuan Tan, Yifan Jia, Mianyang Yu, Zichen Song, Xiaokang Peng, Ziyang Xu, Yongxin Ni, et al. 2024. A comprehensive review of community detection in graphs. *Neurocomputing* 600 (2024), 128169.
- [36] Ma Li, Long Hunter, , Liu Rhea, Zou Mia, Wang Vik, Wang Lois, and Li Dannie. 2017. 2017 WeChat user and business ecosystem report. <https://wechatwiki.com/wp-content/uploads/2017-WECHAT-USER-AND-BUSINESS-ECOSYSTEM-REPORT.pdf>. Accessed: 2025-04-09.
- [37] Rich Ling, Troels Fibæk Bertel, and Pål Roe Sundsøy. 2012. The socio-demographics of texting: An analysis of traffic data. *new media & society* 14, 2 (2012), 281–298.
- [38] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. 2007. l-diversity: Privacy beyond k-anonymity. *Acm transactions on knowledge discovery from data (tkdd)* 1, 1 (2007), 3–es.
- [39] Nayantara Mallesh and Matthew Wright. 2007. Countering statistical disclosure with receiver-bound cover traffic. In *Computer Security—ESORICS 2007: 12th European Symposium on Research In Computer Security, Dresden, Germany, September 24–26, 2007. Proceedings* 12. Springer, 547–562.
- [40] Nick Mathewson and Roger Dingledine. 2004. Practical traffic analysis: Extending and resisting statistical disclosure. In *International workshop on privacy enhancing technologies*. Springer, 17–34.
- [41] Ulf Möller, Lance Cottrell, Peter Palfrader, and Len Sassaman. 2003. Mixmaster Protocol – Version 2. IETF Internet Draft.
- [42] Christian Montag, Konrad Blaszkiewicz, Rayna Sariyska, Bernd Lachmann, Ionut Andone, Boris Trendafilov, Mark Eibes, and Alexander Markowetz. 2015. Smart-phone usage in the 21st century: who is active on WhatsApp? *BMC research notes* 8 (2015), 1–6.
- [43] Shishir Nagaraja. 2007. Anonymity in the wild: Mixes on unstructured networks. In *International workshop on privacy enhancing technologies*. Springer, 254–271.
- [44] Mark EJ Newman and Juyong Park. 2003. Why social networks are different from other types of networks. *Physical review E* 68, 3 (2003), 036122.
- [45] Aleksander Øhrn and Lucila Ohno-Machado. 1999. Using Boolean reasoning to anonymize databases. *Artificial Intelligence in Medicine* 15, 3 (1999), 235–254.
- [46] Simon Oya, Fernando Pérez-González, and Carmela Troncoso. 2016. Design of pool mixes against profiling attacks in real conditions. *IEEE/ACM Transactions on Networking* 24, 6 (2016), 3662–3675.
- [47] Simon Oya, Carmela Troncoso, and Fernando Pérez-González. 2014. Understanding the effects of real-world behavior in statistical disclosure attacks. In *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 72–77.
- [48] Fernando Pérez-González, Carmela Troncoso, and Simon Oya. 2014. A least squares approach to the static traffic analysis of high-latency anonymous communication systems. *IEEE Transactions on Information Forensics and Security* 9, 9 (2014), 1341–1355.
- [49] Andreas Pfitzmann and Marit Hansen. 2005. Anonymity, unlinkability, unobservability, pseudonymity, and identity management—a consolidated proposal for terminology. (2005).
- [50] Dang Vinh Pham and Dogan Kesdoğan. 2015. Towards relations between the hitting-set attack and the statistical disclosure attack. In *IFIP International Information Security and Privacy Conference*. Springer, 35–50.
- [51] Ania M Piotrowska, Jamie Hayes, Tariq Elahi, Sebastian Meiser, and George Danezis. 2017. The loopix anonymity system. In *26th usenix security symposium (usenix security 17)*. 1199–1216.
- [52] Fabian Poignée, Anika Seufert, Frank Loh, Michael Seufert, and Tobias Hofffeld. 2024. Fitting the Puzzle: Towards Source Traffic Modeling For Mobile Instant Messaging. In *2024 15th International Conference on Network of the Future (NoF)*. IEEE, 229–237.
- [53] Javier Portela, Luis Javier García Villalba, Alejandra Guadalupe Silva Trujillo, Ana Lucila Sandoval Orozco, and Tai-Hoon Kim. 2016. Disclosing user relationships in email networks. *The Journal of Supercomputing* 72 (2016), 3787–3800.
- [54] Avi Rosenfeld, Sigal Sina, David Sarne, Or Avidov, and Sarit Kraus. 2016. WhatsApp usage patterns and prediction models. In *ICWSM/IUSSP Workshop on Social Media and Demographic Research*.
- [55] Avi Rosenfeld, Sigal Sina, David Sarne, Or Avidov, and Sarit Kraus. 2018. WhatsApp usage patterns and prediction of demographic characteristics without access to message content. *Demographic Research* 39 (2018), 647–670.
- [56] Paul Rösler, Christian Mainka, and Jörg Schwenk. 2018. More is less: On the end-to-end security of group chats in signal, whatsapp, and threema. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 415–429.
- [57] Marc Roßberger and Doğan Kesdoğan. 2023. Smart noise detection for statistical disclosure attacks. In *Nordic Conference on Secure IT Systems*. Springer, 87–103.
- [58] Marc Roßberger and Doğan Kesdoğan. 2026. Herausforderungen bei der Anonymisierung von Instant Messaging: Eine datengestützte Analyse. In *SICHERHEIT 2026*. Gesellschaft für Informatik, Bonn, 10–18420.
- [59] Ebrahim Sahafizadeh and Behrouz Tork Ladani. 2020. A model for social communication network in mobile instant messaging systems. *IEEE transactions on computational social systems* 7, 1 (2020), 68–83.
- [60] Andrei Serjantov and George Danezis. 2002. Towards an information theoretic metric for anonymity. In *International Workshop on Privacy Enhancing Technologies*. Springer, 41–53.
- [61] Anika Seufert, Carina Baur, Fabian Poignée, Michael Seufert, and Tobias Hofffeld. 2024. Sitting, Chatting, Waiting: Influence of Loading Times on Mobile Instant Messaging QoE. In *2024 16th International Conference on Quality of Multimedia Experience (QoMEX)*. IEEE, 100–103.
- [62] Anika Seufert, Fabian Poignée, Michael Seufert, and Tobias Hofffeld. 2023. Share and multiply: modeling communication and generated traffic in private WhatsApp groups. *IEEE Access* 11 (2023), 25401–25414.
- [63] Michael Seufert, Tobias Hofffeld, Anika Schwind, Valentin Burger, and Phuoc Tran-Gia. 2016. Group-based communication in WhatsApp. In *2016 IFIP networking conference (IFIP networking) and workshops*. IEEE, 536–541.
- [64] Vivian C Sheer and Ronald E Rice. 2017. Mobile instant messaging use and social capital: Direct and indirect associations with employee outcomes. *Information & Management* 54, 1 (2017), 90–102.
- [65] Signal Foundation, Signal Messenger LLC and contributors. 2025. *Signal*. <https://signal.org>
- [66] Latanya Sweeney. 2002. k-anonymity: A model for protecting privacy. *International journal of uncertainty, fuzziness and knowledge-based systems* 10, 05 (2002), 557–570.
- [67] Ying Tang and Khe Foon Hew. 2017. Is mobile instant messaging (MIM) useful in education? Examining its technological, pedagogical, and social affordances. *Educational Research Review* 21 (2017), 85–104.
- [68] Ramroop Tara. 2020. *A 2021 guide to live chat support*. Retrieved 2025-03-05 from <http://web.archive.org/web/20080207010024/http://www.808multimedia.com/winnt/kernel.htm>
- [69] International Telecommunication Union. 2002. *ITU-T Recommendation G.1010 - End-user multimedia QoS categories*. Retrieved 2025-03-05 from <https://www.itu.int/rec/t-rec-g.1010-200111-i>
- [70] Jelle Van Den Hooff, David Lazar, Matei Zaharia, and Nickolai Zeldovich. 2015. Vuvuzela: Scalable private messaging resistant to traffic analysis. In *Proceedings of the 25th Symposium on Operating Systems Principles*. 137–152.
- [71] Duncan J Watts and Steven H Strogatz. 1998. Collective dynamics of ‘small-world’ networks. *nature* 393, 6684 (1998), 440–442.
- [72] WhatsApp. 2024. *WhatsApp Encryption Overview - Technical white paper*. Retrieved 2025-03-05 from <https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf>
- [73] David Isaac Wolinsky, Henry Corrigan-Gibbs, Bryan Ford, and Aaron Johnson. 2012. Dissent in numbers: Making strong anonymity scale. In *10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*. 179–182.
- [74] David Isaac Wolinsky, Ewa Syta, and Bryan Ford. 2013. Hang with your buddies to resist intersection attacks. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. 1153–1166.
- [75] Zhen Xiao, Lei Guo, and John Tracey. 2007. Understanding instant messaging traffic characteristics. In *27th International Conference on Distributed Computing Systems (ICDCS’07)*. IEEE, 51–51.
- [76] Peng Zhou, Xiapu Luo, and Rocky KC Chang. 2013. More anonymity through trust degree in trust-based onion routing. In *Security and Privacy in Communication Networks: 8th International ICST Conference, SecureComm 2012, Padua, Italy, September 3-5, 2012. Revised Selected Papers* 8. Springer, 273–291.
- [77] zzz (Pseudonym) and Lars Schimmer. 2009. Peer Profiling and Selection in the I2P Anonymous Network. In *Proceedings of PET-CON 2009.1* (Dresden, Germany). 59–70.