The Disadvantages of Free MIX Routes and How to Overcome Them

Oliver Berthold[1], Andreas Pfitzmann[1], and Ronny Standtke[2]

[1] Dresden University of Technology, Germany
{ob2,pfitza}@inf.tu-dresden.de
[2] Secunet, Dresden
Ronny.Standtke@gmx.de

Abstract. There are different methods to build an anonymity service using MIXes. A substantial decision for doing so is the method of choosing the MIX route. In this paper we compare two special configurations: a fixed MIX route used by all participants and a network of freely usable MIXes where each participant chooses his own route. The advantages and disadvantages in respect to the freedom of choice are presented and examined. We'll show that some additional attacks are possible in networks with freely chosen MIX routes. After describing these attacks, we estimate their impact on the achievable degree of anonymity. Finally, we evaluate the relevance of the described attacks with respect to existing systems like e.g. Mixmaster, Crowds, and Freedom.

1 Introduction

The concept of MIXes was developed in 1981 by David Chaum [3], in order to enable unobservable communication between users of the internet. A single MIX does nothing else than hiding the correlation between incoming and outgoing messages within a large group of messages.

If participants exclusively use such a MIX for sending messages to each other, their communication relations will be unobservable - even though if the attacker does control all the network connections. Without additional information not even the receiver gets to know the identity of the message's sender.

When using only one MIX, one has to rely upon its security completely. Therefore usually several MIXes are used in a chain. Now, any single MIX does not have all the information which is needed to reveal communication relations. At worst, a MIX may only know either sender or receiver.

According to the attacker model for MIXes, the communication relations have to be kept secret even in case that all but one MIX cooperate with an attacker who taps all lines (called *global attacker*). But a sufficient number of reliable participants is necessary: A single participant can not be anonymous if all or most of the other participants are controlled by the attacker. All possible attacks on a MIX network are to be examined with regard to that attacker model.

H. Federrath (Ed.): Anonymity 2000, LNCS 2009, pp. 30–45, 2001.

There are different possibilities to organise the co-operation of several MIXes. Generally, all MIXes exist independently from each other in the internet. When anybody wants to use a certain MIX, he simply sends his message to it, respectively, he has another MIX send his message to it. This kind of co-operation is further called a *MIX network*.

A special variation is to define a single valid chain of MIXes for a group of participants. This we will call a *MIX cascade* (compare [6]).

Apart of those two concepts there is a high variety of hybrid combinations of MIX network and MIX cascade. For example, possible configurations are:

- **A network of cascades:** The sender chooses a number of defined cascades through which his message is sent sequentially.
- **A network with restricted choice of patches:** Every MIX defines its possible successors to send messages to.
- **Multiple-duplicated cascades:** For every position within the cascade several physical MIXes are defined, such that a user can choose stepwise.
- **Tree structures:** Here a MIX cascade is used which has on some positions duplicated MIXes. A possible scenario may consist of many first-step MIXes and a single last-step MIX. Now, a distribution and acceleration can be reached when the anonymity group is to be quite large and not all the dummy traffic is to be forwarded to the last MIX.

Up to now these combinations were examined only partially. But in contrast to the MIX network some kinds of attacks which we will describe later in this paper, are not possible or may be prevented more easily. We will discuss the advantages of different configurations when describing types of attacks.

As the headline says, compare the MIX cascade with the MIX network with regard to:

- Attacks being possible under one configuration but not under the other.
- Attacks that can only be prevented under one of both configurations (- nowadays).
- Evaluation about the qualification of MIX network and MIX cascade to guarantee anonymity and unobservability with a global attacker as he was described above.

First of all we will check how unobservability and anonymity can be reached by the use of MIXes and how it is possible to evaluate anonymity. In the following section will be examined what kinds of attacks exist in general and how they can be prevented. The final section then deals with the question how these attacks may endanger existing anonymity services.

2 Unobservability and Anonymity by MIXes

As already said in the introduction, a MIX hides the relation between the incoming and outgoing messages. This is done by collecting a number of messages and reorder them before sending them on their way.

Because a particular outgoing message could have been sent by any of the senders of the incoming messages, the sender of this message is unobservable within that group. But on the other hand, it is a fact that the message was certainly sent from within that group. The degree of anonymity can be defined by the size of the group, i.e. the number of possible senders. For example, the anonymity may be measured as

$A = ld(n)$ [bit] where n is the number of senders. Its meaning is the logarithm with base 2 of n.

An attacker who wants to find out the sender of a particular message does reach his aim with the same probability as he may guess the value of a random string of A bits.

When using a MIX cascade, the degree of anonymity stays the same if all messages received by the first MIX are forwarded correctly to the last MIX. This is true even if only one MIX in the cascade doesn't work together with the global attacker. However, the degree of anonymity decreases when messages are lost on their way through the cascade (e.g. because of active attacks on the link between sender and first MIX of the cascade). It decreases in the same amount as the number of those senders decreases whose messages are still existing.

The situation is different in a MIX network. When a MIX is directly receiving messages from a set of MIXes the anonymity group of senders is the union of the anonymity groups of all these MIXes.

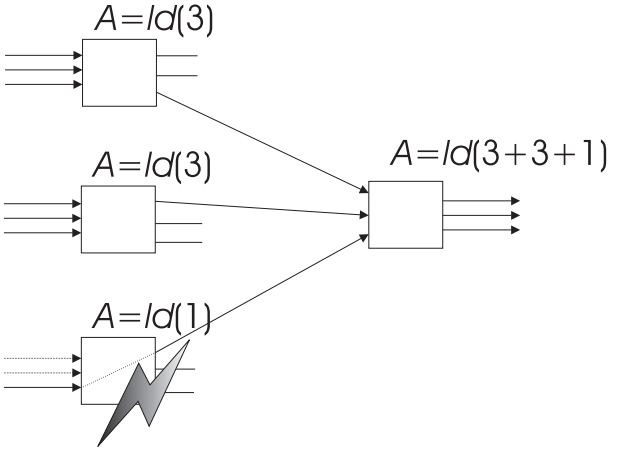

Fig. 1. Anonymity group in a MIX network

Of course, that is only true if all the MIXes are trustworthy. If an attacker is getting to know the way of reordering in a MIX, this particular MIX doesn't contribute any longer to the degree of anonymity. That means, outgoing messages

from this MIX have an anonymity group of n=1, if this MIX is the first or last one or if the attacker controls all MIXes before or behind it.

Another problem is arising for all combinations of MIXes, when the attacker is able to recognise, that several messages sent at several times can be linked to one and the same sender.

If the participants join and leave the group from time to time, the anonymity of a group of linkable messages is only among those senders who were part of the group for the whole time.

An attacker may easily be able to link messages when the senders do not communicate with each other but rather want to use public services like internet anonymously. The connection to the internet server is usually not encrypted and hold up for a certain time, during which several packets of data are exchanged.

In order to make intersections of anonymity groups more difficult often dummy traffic is used which can not be distinguished from real messages. That proceeding ensures that every active user always is part of the anonymity group.

3 Attacks on a Single MIX

3.1 Message Size

The attacker could distinguish the messages sent by a MIX, if they would be different in size. Because the size of the sent message is the same as the size of the received message, an attacker could correlate them. A solution is, that every message should have exactly the same size. If the data is less than this defined message size, padding is used. In a MIX network each message contains information special for a certain MIX. So this information is extracted from the message and the MIX has to fill the free space using padding, so that the message size remains the same.

This is not necessary at all in a MIX cascade, because all messages take the same way and are shortened to the same size each time.

3.2 Replay

The decryption of a message done by each MIX is deterministic. So if the same message would pass a MIX, the result of the decryption sent by the MIX would also be the same. An attacker who observes the incoming and outgoing messages, could send the same message to the MIX again, and would see which message is sent twice from the MIX. So this MIX is bridged and the conclusion is, that a MIX must not process messages he has already processed.

Usually, this is achieved by using a database in which a MIX stores every processed message. If a new message arrives, he first checks if this message isn't already stored in this database.

3.3 Manipulating of Messages

If an attackers changes only some bits of a message, then if inappropriate cryptography is used he can detect this broken message also after it has passed a trustworthy MIX. This is possible especially if:

- he is the receiver of the message, or
- he controls the server, which would send the decrypted message to the real receiver, or
- a MIX transforms the message in such a way, that the manipulated bit(s) would remain on the same position (i.e. if the MIX uses the stream cipher modus OFB). Assuming that there are a trustworthy MIX followed by an attacking MIX. The attacker changes a bit of that part of the message, which belongs to the message header for the attacking MIX. Now he sends this message to the trustworthy MIX, which will forwards the message to the attacking MIX. If this MIX decrypts the message, he would detect the error in the message header, but restores it (changes the bit again). The attacker now knows the relation between the manipulated incoming message and a correct outgoing message and can trace it to the receiver.

The attacker detects the broken message, because the final decrypted message (which goes to the receiver) usually contains a lot of redundancy or the message header for a MIX has to satisfy a well known format.

To prevent these attacks, each MIX has to verify the integrity of every received message by checking included redundancies. This could be a message hash generated by the sender and included in the message header for each MIX.

3.4 Blocking of Messages

If an attacker blocks some messages, the senders of these messages are excluded from the anonymity group. The same result is achieved, if some messages are manipulated (as described above).

The extreme case is known as "n-1 attack". In this case all messages but one, will be blocked, manipulated or generated by the attacker. So the only message the attacker doesn't know is the message he wants to trace. So the attacker has bridged the trustworthy MIX or has limited the anonymity group of the remaining messages (if there are more than one).

There exists no general applicable method, in order to prevent this attacks. A possible solution is that the MIX must be able to identify each user (sender of messages). The MIX has ensure that the messages he receives are sent by enough different users and so the attacker doesn't control a majority of them.

The next section discusses how this could be done by the MIXes, considering the different possibilities of configuration (network or cascade).

Under certain circumstances it is possible that an attacker could uncover the route of a message, if he only blocks a single message:

If the attacker can find out the plaintext of messages and the user sends many messages, which definitely are related (i.e. the blocked message is part of a bigger data stream) then the attacker could detect in which data stream the message he blocked is missing.

4 Comparison MIX Network - MIX Cascade

4.1 Motivation

There still exists a controversial discussion, which kind of a MIX configuration is the better one. The advocates of the MIX-network are always pointing out that there are the following advantages:

Each user can decide on his own, which MIXes he wants to trust. Because an attacker has the best chance to observe a user, if he controls as many MIXes as possible (or even all MIXes), it seems that a MIX cascade is extremely unsuitable.

1. In this case an attacker knows exactly which MIXes he has to control in order to observe a user successfully. But if the user can choose his own route, he can exclude MIXes, which seem untrustworthy to him and an attacker has to control much more MIXes in order to achieve the same probability to attack successfully. The community of users could publish which MIXes are not trustworthy and so would prevent attacks.
2. Because a MIX network could theoretically grow up to an infinite number of MIXes, the anonymity group could also raise to an infinite size. Every user can choose an arbitrary route of MIXes and so an observer could not detect, messages of which users a MIX currently processes.
3. There exists no structure, so a MIX network is very flexible, scalable and extendable. It could be realised as a fault-tolerant network. The only thing, that must be done, is to publish, when a MIX is inserted into or removed from the network. Such a system would be predestined for usage in the Internet.

The advocates of the MIX cascade oppose, that the security of the MIX network can't really be proven up to now. For instance the facts of the 2nd point - as explained above - are only true, if all MIXes are trustworthy. When assuming an attacker model, where only one MIX of a route is trustworthy, the anonymity group wouldn't become bigger than the batch size of this MIX.

In the next section we will describe additional attacks and show, that the level of anonymity is much smaller in reality. Nevertheless, a MIX network is much more flexible compared to a MIX cascade. But it's possible to construct a fault-tolerant MIX cascade, too [5]. And there would be more than one cascade world-wide, so the advantage described in point 1 could be applied to cascades as well: a user chooses the cascade, he trusts. Adding new cascades extends the system as far as needed (Point 3). Compared to the MIX network, an unbalanced load sharing wouldn't have negative impacts on the performance. Only users of a cascade, which is insufficiently used compared to its capacity, would achieve a lower level of anonymity. If the load of a MIX network is non-uniform, then on the one hand some MIX could be temporarily overloaded. On the other hand a MIX with little load would have to wait for a long time to gather enough incoming message or would send many dummies, which would produce additional traffic.

The most important reason, why you shouldn't use a MIX network is the low level of security. This is the conclusion of the authors of this paper, after they found a number of attacks against a MIX network. The following sections describes these attacks and show why they occur especially in a MIX network and why it's hard to prevent them.

4.2 Achievable Level of Anonymity

If we assume that there exists a single trustworthy MIX only, then the achievable level of anonymity could not be higher than the size of the batch of this MIX.

In the following sections we describe attacks, which parts this input batch and so decreases the size of the anonymity group.

Position in MIX route. Based on our attacker model, each user should be anonymous, even if his messages only pass one trustworthy MIX. In a MIX network a certain MIX may be on different positions on the individual routes of the messages he receives. In comparison a MIX of a cascade will always have the same position because the route is static as long as the cascade exists.

The question is: is a MIX of a MIX network able to detect his position in a route of a certain message and if so, can he use this knowledge to partition the input batch?

If the attacker controls all other MIXes, which this message passes through, then it's easy to detect the position of the trustworthy MIX. He simply counts the number of passed MIXes for every message.

Now the attacker knows the composition of the incoming batch: He knows the routing position for each user's message.

If we assume, that every user sends his messages using the same number of MIXes (e. g. sets the route length to the maximum allowed for by the message format used in order to get maximal anonymity), then the attacker also knows, how many MIXes each user's message yet already passed.

Because he controls all following MIXes of each route, he can determine, how many MIXes each sent message has yet to pass. Because all routes have the same length, incoming messages can only have been transformed into outgoing messages if the route lengths would match.

Eventually, a message is only unobservable in that group of messages which have this MIX on the same routing position. So the attacker successfully partitioned the input batch and decreased the level of anonymity of each user. Figure 2 illustrates this attack.

In order to prevent this attack, each user has to choose a different route length for each of his/her messages or they only use each MIX for a particular position within routes. But this particular position within routes would be in contrast to the flexibility of the MIX network and in fact results in a MIX cascade with possibly duplicated MIXes.

The first method, that the user should choose the length of the route, would only partly be a solution: There still exists a constant maximum of the length

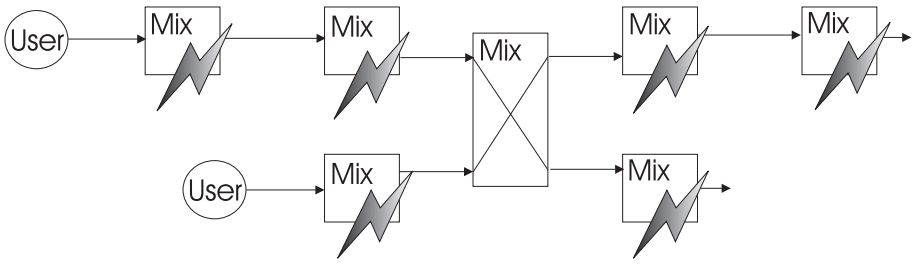

Fig. 2. Routing position of a trusted MIX

of a route, and so it's possible to calculate intersections, using the procedure explained above.

The only advantage is that a bigger number of neighbouring route positions are building a common anonymity group. Nevertheless, the probability that there are different anonymity groups in one batch is still very high. Building only one anonymity group is only achievable, if the route could have an infinite length, because in this case every message could still pass any number of MIXes. But because a message header is needed for each MIX and all messages should have the same size, there always exists a maximum length of routes.

Additionally, it may be possible, that the attacker knows the plaintext of the message. In this case the attacker could calculate how many MIXes a message could have passed at most, even if the quota of the message header is different in order to allow different route lengths. To prevent this, each message header should have the same size. But this implies another restriction on the maximum of the length of a route, because a message should be able to transport as many data as possible and so a short message header, allowing only short routes, would be chosen.

If an attacker can assume, that a user would normally send more than one message to a certain receiver, then the probability of calculating small intersections increases.

If the user chooses a different MIX route for each of these messages, different anonymity groups arise each time. So the attacker could calculate the intersection of all these anonymity groups. Even if all users would always choose the same route, the anonymity groups wouldn't be identical, because the MIX network doesn't work synchronously.

Conclusion: If only one MIX of a route is trustworthy, then the achievable anonymity is distinctly lower in a MIX network compared to a synchronously working MIX cascade.

Using a configuration of MIXes which allows only a fixed routing position for each MIX for all messages it processes prevents the partitioning of the batches and so prevents this attack.

MIX cascades, duplicated MIX cascades and tree structures as explained in the introduction are working in that way.

Determining the next MIX. The previous section describes an attack which is especially successful, if the users select different routes for each message. Now we want to explain an attack, which could be done, if the users send all messages using the same individual route.

A trustworthy MIX of a MIX network receives messages sent by users and other MIXes. The MIX forwards these messages either to users or to other MIXes.

An attacker knows the senders of all messages, which came directly from users. Additionally, he knows all senders of messages which passed only attacking MIXes.

Because a MIX forwards the messages only to some other MIXes or users, only they could be possible receivers of an incoming message.

If there are only a few trustworthy MIXes, only those users which are connected by the described graph, are members of the anonymity group of a received message. This is illustrated in Figure 3.

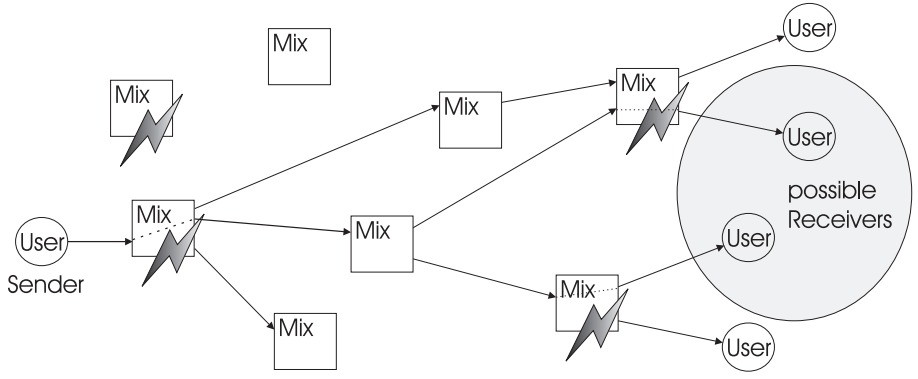

Fig. 3. Intersection set of anonymity group

If the users send their messages, using always the same route, an attacker only needs to observe a trustworthy MIX at those times, when this MIX forwards a message of a sender the attacker is interested in.

A MIX network doesn't work synchronously, so the batches always generate different anonymity groups. So if the attacker would calculate the intersection of these anonymity groups, then after a finite time he knows which MIX has received the message, he is interested in.

The attacker repeats this procedure sequentially attacking the following MIX on the route and so on until he attacks the last MIX.

So the anonymity group of the sent message would become the group of receivers addressed from the last trustworthy MIX on the route. It is assumed, that for all messages there exists a maximal route length known by the attacker.

If a user sends more than one message to the same receiver, then the attacker has a good chance to determine this receiver.

If the attacker is able to observe the whole network and could store all communication relations of all MIXes for a long period of time, he could attack all users and all trustworthy MIXes simultaneously.

A possible solution would be that each MIX has to send at least one message to each other MIX. These messages could be dummies.

If the number of MIXes increases, many dummies have to be sent. All these dummies have to pass more than one MIX, because if they would be sent only to the next MIX, this MIX could be an attacking one and detect that this message was only a dummy. The problem is that all these dummies would dramatically increase the network traffic.

On the other hand, the achievable security is uncertain: Based on the attacker model only one MIX of a route is trustworthy, so that the following variations are possible:

- *The generator of dummies is a trustworthy MIX:* All other MIXes of the route are controlled by the attacker, so he knows which messages are dummies.
- *The trustworthy MIX is somewhere else on the route:* The attacker controls the generator of dummies, so he knows which MIX gets dummies and which one gets real messages.

So the attacker would detect to which MIX or participant the observed MIX forwards the received messages, although dummies were sent.

We think, that sending dummies in this way isn't practicable.

Another solution for this problem could be, that the user has to choose his MIX route with regard to some rules.

For each MIX, some neighbouring MIXes are defined. So the MIX has to send dummies only to these MIXes, what reduces the expenditure. In this case the explained attack wouldn't be possible.

We described this configuration in the introduction as an alternative to the completely freely chosen MIX network. Other configurations are possible, too - like tree structured networks, with the last MIX as root or a MIX cascade. They would prevent the attack, because each MIX has only one successor.

4.3 Probability of Unobservability

If we use a MIX cascade, then we optimize for the case that only one MIX is trustworthy. Based on the functionality of each MIX, also in this case the transported messages are unobservable, at least if no active attacks occur.

In the following section, we calculate the probability, that at least one MIX of a randomly selected route of a MIX network is trustworthy.

Because each user chooses only some MIXes for his route, the probability that he chooses at least one trustworthy MIX is:

$$p = 1 - a^l$$

(a is the quota of attacking MIXes and l is the length of the route)

This formula is only true, if each MIX could be selected more then once for each route. But this isn't very clever, because if a MIX is selected repeatedly, then the probability decreases, that at least one trustworthy MIX is passed by the message. The conclusion is that each message should pass each MIX at most once. For this case the formula for calculating the probability p is as follows:

$$p = 1 - \frac{M_A! \cdot (M_A + M_G - l)!}{(M_A - l)! \cdot (M_A + M_G)!}$$

(M_A is the number of attacking MIXes, M_G is the number of trustworthy MIXes and l is the length of the route.)

If a user during the time sends many messages, which belong together (this means, that an attacker knows, that these message are sent from the same user), then we have to raise the probability p to the power of the number of messages k. So the probability to observe a user is:

$$1 - p^k$$

Example (comparison MIX cascade - MIX network):
length of cascade (route): 4 MIXes

Based on our attacking model 3 MIXes of this cascade could be attackers. That are 75% attacking MIXes.

If we assume that 20 MIXes are building a MIX network and that 75% of these MIXes are attacking ones, too, then a randomly chosen route contains at least one trustworthy MIX with a probability of 71.8%.

4.4 Active Attacks

In Sections 3.3 and 3.4 we described active attacks on a single MIX. We showed that an attacker who manipulates messages (Section 3.3) wouldn't be successfully, if each MIX verifies the integrity of each message.

But this solution nevertheless doesn't prevent all effects of this attack. It only changes this attack in an attack, which blocks messages (Section 3.4) because broken messages would be deleted.

It's not easy to prevent these message blocking attacks and especially these attacks are very dangerous:

If the attacker knows the plaintext of the message, then he can use active attacks, in order to achieve the following:

- If he blocks the messages of some users, he can decrease the anonymity group of a certain user. If he can also link some messages of this user, then blocking only few messages could be enough to detect the sender.
- The attacker may detect, that he doesn't receive a message, he is waiting for. This is possible if this message is part of a larger data stream, that would be broken without this message. So in an extreme case blocking one message could be enough to uncover the communication relation.

A possible solution to prevent these active attacks, results in deleting all messages of the whole MIX network if one is blocked.

A fundamental problem in detecting these attacks is the following:

If an attacker only blocks some messages, a MIX would wait until he gets enough messages. So an attacker will generate some own messages and will send them to the MIX.

How should a MIX decide, if an incoming message comes either from the attacker or from a real user? He has to know this, in order to recognise the attack.

There exist several different basic approaches for the MIX cascade, in order to solve this problem.

Users verifying the MIXes. In this case all messages a MIX receives are sent to all users. So each user can check that his message has arrived at the MIX. If that's true, he sends an acknowledgement to the MIX. The MIX forwards the batch only to the following MIX, if he receives these acknowledgements from all users.

The problem is, that the MIX must know, which users have sent messages and the MIX has to ensure, that a certain acknowledgement was really sent by one of these participants.

So each user has to own a digital certificate and authenticates himself (and the acknowledgement he has sent) to the MIX.

In the case of a MIX network, this could be a problem, because:

- The attacker shouldn't know, who has sent a message to which MIX, because this would extremely reduce the achievable anonymity and the attacks explained in Section 4.2 could be done much easier. In order to prevent this, each user has to send acknowledgements to all MIXes of the network, even if he hasn't used them. And he also has to receive all messages published by each MIX. Of course this would dramatically decrease the efficiency.
- But now an acknowledgement sent by a user doesn't mean in every case that one of his messages is currently processed and so the acknowledgements tell nothing about the number of real messages in a batch. When using big MIX networks there always exists the risk that the anonymity group could be very small.
- A MIX network doesn't work synchronically, so the user has to decide, if his message has been blocked or if it only arrived late for being processed in the current batch. A successful attack could be to delay $n - 1$ messages for one batch.

Ticket-method. A new idea developed by our group is, that the MIX only accepts messages which come verifiable from the group of registered participants. Each registered participant gets exactly one ticket from every MIX, which is only valid for a certain batch. If a MIX processes a batch, he verifies, that every message contains such a valid ticket. Because each participant gets only one

ticket for each batch, he can only send one message. So the attacker could only generate as many messages as the number of users he controls. A MIX couldn't link a ticket to the belonging user, because we use blind signatures. The whole procedure is explained in detail in [2] (contained in this book).

This procedure is only badly applicable in a MIX network, because:

- The user has to request tickets from every MIX in the world, whereas in a cascade, the user only has to request tickets from each MIX in the cascade he/she intents to use.
- The user doesn't know exactly, which batch will contain his message. Therefore, the user has to request tickets from every MIX for many batches and include many tickets for each step.
- And finally only a part of these ticket would be used so that a MIX could only ensure, that no additional messages sent by an attacker would be processed. But the attacker could decrease the anonymity group by blocking some messages. in the cascade, the number of tickets issued could exactly match the batch size.

Summarising, the described methods to prevent active attacks are very expensive. A user of a MIX network couldn't use every MIX at every time, so:

- the expenditure grows up excessively, because nevertheless, the user has to execute these methods with every MIX.
- These procedures wouldn't achieve an acceptable level of security

As a result, we think, that it isn't possible to protect a MIX network against such attacks with justifiable expenses. So an active attacker could easily observe at least a certain user.

5 Classification of Existing Architectures

In this section we discuss, how the described attacks are related to existing and used services.

But we have to say that these services are developed assuming that there are only substantial weaker attackers in comparison to our assumption about strong attackers.

5.1 Remailer

Anonymous remailers, like for instance Mixmaster have the advantage that they need not to work real-time.

Because of that Remailer-MIXes could use a different method for building an anonymity group. They collect incoming messages in a pool. Outgoing messages are selected casually out of this pool. Because a message theoretically can remain in the pool for an unlimited amount of time, none of the already received messages can be excluded of the anonymity group. Because of that the described

attacks (Section 4.2) that limit the size of the anonymity group can rarely be used successfully.

It can be assumed that the messages are atomic and unlinkable because the recipients are many different users, who can be trusted in most cases according to the attacker model. But the attacks as described in Section 4.2 and 4.4 are especially successful if messages can be linked. Attacks by using the length of messages (Section 3.1) and the replay of messages (3.2) are at least prevented by the Mixmaster. The attack by manipulating messages (3.3) is not possible because it is assumed that the attacker gets the plaintext of messages.

It is possible to block messages and a MIX can be flooded with messages by the attacker because the remailer can not check the origin of a message (4.4). The attacker doesn't know the number of messages stored in the pool, because the delay of a message in the MIX is unknown for it.

This attack for remailers is only possible with a certain probability in contrast to MIXes that work in a deterministic way. It increases with expenditure. The attack by choice of MIX routes (4.3) can't be prevented, because it is a net of remailers similar to the MIX network.

Result: Some of the described attacks are theoretically possible. Because the remailers don't work in a deterministic way the attacks are complicated. Because of that pool mixes cannot be used for real-time services. The Mixmaster remailer in comparison to other anonymity services as described in Section 5 offers the best protection against observation.

5.2 Onion-Routing/Freedom

Zero Knowledge offers the commercial system "Freedom" [4]. This is a MIX network that provides unobservable and anonymous real-time connections between network nodes. So it provides access to different internet services. The way how it works is very similar with the service Onion-Routing [7].

To achieve little delay for every user an anonymous route across the MIX network is provided. The advantage of this routes is that the data must only be encrypted or decrypted via a symmetrical cryptographic system. Of course all data transmitted over one route are linkable. Because these routes can be generated every time and the users send different amounts of data at different times via the route an attack by using the length of messages (3.1) is possible.

Also a replay attack (3.2) is possible. There is no checking of message replay in the MIXes. Instead of that Freedom uses link encryption between neighbouring MIXes.

Because of that an external attacker that controls none of the MIXes can't achieve much:

- Via the use of streaming cipher even identical messages result in a different representation of bits.
- The external attacker can't add messages because he does not know the key needed.

– Because of dummy traffic there exists a permanent traffic load between the mixes. That prevents successful attacks by traffic analysis.

But if the attacker controls part of the MIXes specific attacks via message blocking (3.4, 4.4) or traffic analysis (4.2) are possible: an attacking MIX knows its own key used for link encryption and can divide dummy traffic from real messages.

The Freedom network only accepts registered users, that identify themselves via tokens. This check is only done by the last MIX of the chosen route. Because of this the attacker can generate messages on its own, which are accepted by MIXes.

Even if only authenticated messages would be accepted by MIXes this attack could not be prevented because ever user theoretically can own an unlimited amount of tokens.

The attack explained in Section 4.2.2 which would determine the route of a message is only partly realisable. To each MIX per definition belong some neighbouring MIXes. This decreases the number of possible routes and this in fact increases the size of the anonymity group.

Each MIX reorders the messages only in a very limited amount. So if an attacker controls neighbouring MIXes, the anonymity groups would become very small.

Conclusion: The Freedom network only prevents attacks from external observers and isolated attacking MIXes. If some MIXes of a route cooperate with the attacker, the attacker has a very good chance to observe a user.

5.3 Crowds

The Crowds system is based on a very different principle. A participant sends a message only with a certain probability directly into the internet. Otherwise he forwards the message to an other randomly selected user. This user does the same and so on.

A message passes several users and all these users get knowledge of the plaintext. An external attacker wouldn't get the plaintext, because the connections between the users are encrypted.

Based on the attacking model explained above, this isn't a problem, because most participants are trustworthy ones. Eventually no user knows if a request he receives from another user really comes from this user.

An assumption is that the attacker isn't able to observe the whole network. Otherwise he could detect, if a user receives a message from an other one. If a user sends a message and hadn't received one, this message must come from himself.

Conclusion: Crowds protect only against observations done by the communication partner (i.e. the WEB-server) or done by an other, isolated participant.

The described attacks are not applicable, because Crowds are based on a different principle. On the other hand all attacks assume a much stronger attacker than the designer of Crowds expected.

6 Summary

Based on the described attacks and their high probability of success the Conclusion is that a MIX network can't be realised securely, especially when we think of the well known attacking model defined by David Chaum. The theoretical anonymity group of a very big MIX network, that contains all users, is practically not achievable, even if all MIXes are trustworthy ones.

In Section 4.4 we described attacks, which block messages. These attacks in principle apply to MIX cascades, too. But for this case promising basic approaches have been developed in order to prevent these attacks [1] [2].

If a service is to be constructed, which is secure against global attackers, a MIX cascade or any other of the described configurations should be chosen instead of a MIX network. This would prevent at least some of the illustrated attacks.

A disadvantage of the MIX cascade is that the cascade consists of default MIXes which have to be used. A user cannot express his trust in certain MIXes by using them or his distrust by not using them. But a user may choose that cascade he wants to use and trust in.

References

1. Oliver Berthold, Hannes Federrath, Marit Köhntopp: Project "Anonymity and Unobservability in the Internet"; Workshop on Freedom and Privacy by Design; in: Proceedings of the Tenth Conference on Computers, Freedom & Privacy; CFP 2000: Challenging the Assumptions; Toronto/Canada, April 4-7, 2000; ACM, New York 2000; 57-65; http://www.inf.tu-dresden.de/ hf2/publ/2000/BeFK2000cfp2000/.
2. Oliver Berthold, Hannes Federrath, Stefan Köpsell: Web MIXes: A system for anonymous and unobservable Internet access. Proceedings of Workshop on Design Issues in Anonymity and Unobservability, July 25-26, 2000 in Berkeley.
3. David Chaum: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of the ACM 24/2 (1981) 84-88.
4. The Freedom Network Architecture. Zero-Knowledge-Systems, Inc., 1998. http://www.freedom.net/
5. Andreas Pfitzmann: Diensteintegrierende Kommunikationsnetze mit teilnehmerüberprüfbarem Datenschutz; University Karlsruhe, Department of computer science, Dissertation, Feb. 1989, IFB 234, Springer-Verlag, Heidelberg 1990.
6. Andreas Pfitzmann, Birgit Pfitzmann, Michael Waidner: ISDN-MIXes - Untraceable Communication with Very Small Bandwidth Overhead. Proc. Kommunikation in verteilten Systemen, IFB 267, Springer-Verlag, Berlin 1991, 451-463.
7. M. G. Reed, P. F. Syverson, D. Goldschlag: Anonymous Connections and Onion Routing. IEEE Journal on Selected Areas in Communication. Spezial Issue on Copyright and Privacy Protection, 1998. http://www.onion-router.net/Publications/JSAC-1998.ps