Ph.D.-FSTC-5-2009
The Faculty of Sciences, Technology and Communication

DISSERTATION

Defense held on 09/06/2009 in Luxembourg

in candidature for the degree of

DOCTOR OF THE UNIVERSITY OF LUXEMBOURG

IN COMPUTER SCIENCE

by

Volker FUSENIG
Born on 24 April 1981 in Trier, Germany

ANONYMITY AND UNLINKABILITY IN ELECTRONIC COMMUNICATIONS

Dissertation defense committee

Dr Thomas Engel, dissertation supervisor
Professor, Université du Luxembourg

Dr Ulrich Sorger, Chairman
Professor, Université du Luxembourg

Dr Simin Nadjm-Tehrani, Vice Chairman
Professor, Linköping University

Dr Mario Gerla
Professor, University of California, Los Angeles

Dr Uwe Roth
Centre de Recherche Public Henri Tudor

Abstract

Imagine a set of communication partners wants to keep their communication links secret. Consider the case where untrustworthy parties are able to observe every communication, which implies not only that they can detect the content of the communication, but also who is communicating and who is listening. Using this information, the untrustworthy parties try to link communicating parties. This, in a nutshell, is the problem of anonymous and unlinkable communication in computer networks.

By use of encryption techniques the content of messages can be kept private. However, the communication links can still be detected. Since the addresses of sending and receiving parties are contained in the header of every message sent over the network, an untrustworthy party needs only to eavesdrop a single message of the communication in order to link sender and receiver. Additional techniques have to be used to hide this information. We address this problem in this thesis.

We define measures for anonymity and unlinkability that are based on the information theoretic notion of entropy. These measures are used first to evaluate different approaches for anonymous and unlinkable communication and second, to show the effectiveness of attacks on these protocols.

We present existing techniques for anonymous and unlinkable communication and highlight weak points of these techniques by applying attacks to them. In these attacks, known as traffic analysis attacks, the attacker basically tries to collect as much information about the communication as possible and then makes deductions concerning the communication links. We show that these traffic analysis attacks are applicable to many existing techniques. Furthermore, we introduce a new traffic analysis attack, namely the slotted packet-counting attack.

Motivated by these findings, we present a protocol for unlinkable communication in computer networks. We prove that this protocol leaks no information on communication links in the case where attackers are able to observe any communication in the network. By this means, the protocol guarantees a user-defined degree of unlinkability. We also show that the protocol generates a minimal amount of extra messages for achieving a given degree of receiver anonymity, i.e. where an attacker is not able to detect the receiver of a message.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

IP-based communication is used in many different applications. We browse the Internet for news, check the weather forecast or buy a new book at Amazon. We keep in contact with friends or business partners via email, social networks (Facebook, LinkedIn) or Internet telephony (Skype). And finally we update our computer systems over the Internet, load new software or share files online. The common advantage of these applications is that they are fast, convenient and cheap or even free.

The danger of this is that user profiles can be created by combining the information revealed during the use of the different applications. For example an Internet service provider (ISP) is able to observe the communication of its customers and might collect information on the web sites visited. Operators of any website might be able to use cookies to track the communication of users who visited this site. This information can be useful for targeted advertising. The location of a device and its user might be detectable by using geolocation techniques.

Most users do not know that personal information is collected and can be mapped to their identity. Moreover, users often give away personal information for small benefits, such as small discounts when using PayBack cards. Users have two main ways of ensuring that no personal information is misused. In most cases, privacy policies regulate the use of private information by companies. But few users really read these policies; most assume that the company is trustworthy. It is also hard to prove that companies follow these policies. And even if the user can detect that a company misused private information, it is hard to undo the misuse.

The better choice is to only reveal as little private information as possible. By hiding all other information, the users can prevent this information from being misused. While encrypting the communication can guarantee the secrecy of the content communication links can also reveal information, for example that a user has an account at Facebook.

This thesis is about anonymous and unlinkable communication in IP-based networks. By using anonymous communication, i.e. where the sender or the receiver cannot be detected, or unlinkable communication, i.e. where no sender can be mapped to a receiver, the user can secure information on communication links in order to preserve anonymity or unlinkability (definitions see Sections 2.1.2 and 2.1.3).

The rest of the chapter is organized as follows: Section 1.1 gives the problem statement of this thesis. The following Section 1.2 gives an overview of the dissertations' area of study. We show the significance of the study in Section 1.3. The final Section 1.5 gives an overview of this thesis.

1.1 Problem Statement

In IP-based communication, the Internet service provider (ISP) assigns unique IP addresses to its users when they connect to the network. When a user A sends a message M to a station B, the IP addresses of both are part of M. For that reason, all stations that see message M know that the message was sent by A and is addressed to B. Because in general, any two stations on the Internet are not connected by a direct communication link, messages normally travel over a series of stations from the sender A to the receiver B. Consequently, at a minimum all the stations on the communication path learn the IP addresses of A and B.

The mapping from the IP address to the user behind the IP address is known only to the ISP. Therefore the ISP knows to whom A is sending messages. But other parties might also be able to determine the sender and receiver of a communication. In the European Union, the data retention law instructs all member states to store the mapping of users to IP addresses for at least six months for crime prevention and detection. This data can be accessed by governmental institutions. Even if no direct mapping between the user and the IP address is available, information on this mapping can be gained by observing communication over time. If the user logs in to check

email and later visits another website, the same IP address is used. Indeed, tracking of communication processes over time is already achieved by using cookies.

This information is collected for several reasons: Governmental institutions collect information for crime prevention and detection. Private companies legally collect information for market research and advertising reasons. Information may also be collected illegally for advertising, spying and extortion.

The user is not normally able to prevent third parties from collecting private information. However, in most cases policies regulate how this information is collected and how it is used. If the user does not agree to the policy, he can decide to not use the service. Even if he accepts the policy he cannot be sure whether the third party sticks to the policy and that the data is kept secret. The problem is that, once private information has been leaked, it is hard to make sure that all copies have been deleted.

The best way to prevent the misuse of private information is by not giving this information to third parties unless it is really needed. The use of encryption can hide the content of messages, but not the communicating stations: the fact that two stations are communicating with each other also reveals information. Anonymous or unlinkable communication can help to hide this information.

This thesis is about how to guarantee anonymous and unlinkable communication. In a communication setting there exist two kinds of anonymity: Sender anonymity means that no third party is able to detect the sender of a message, while receiver anonymity means that the receiver of a message is not detectable. Unlinkability of sender and receiver is weaker; it says only that the sender and receiver of a message cannot directly be linked. There exist protocols that try to implement anonymity and unlinkability but all suffer from security or scalability issues. In this thesis we present a new protocol for unlinkable communication that offers provable unlinkability. This means that during the communication no information on the communication link is revealed.

1.2 Overview of Dissertation Study

As mentioned in the problem statement, the aim of this thesis is to guarantee unlinkable communication in IP-based networks. The first step is to define the terms *unlinkability* and *anonymity*, then based on these definitions, to introduce measures for unlinkability and anonymity. These measures are important to evaluate the quality of a protocol. On the one hand it is possible to apply these measures to existing protocols, which are already in use. On the other hand, if the best achievable degrees of unlinkability and anonymity are known, new protocols can be optimized with regard to these values. Optimization means first achieving the best possible degree of unlinkability and anonymity, and second reducing the costs of reaching this goal.

By applying these measures we first analyze existing work and outline the weaknesses of the techniques applied. We try to exploit these weaknesses from the attacker's perspective in order to show that they can be used in practice to break the unlinkability and anonymity of the existing protocols.

We apply the measures for anonymity and unlinkability and measure the information that an attacker can obtain by observing the communication. We try to implement a communication protocol that leaks no information on the communication link, i.e. the mapping of sender and receiver of a message. If this is the case, the attacker cannot break the protocol by means of passive observations. The protocol achieves this by building a communication path over different stations where the destination node is placed randomly on the path. Furthermore, the protocol guarantees receiver anonymity in an optimal way in terms of message overhead. We show how to apply the protocol to Internet and ad hoc communication. Compared to existing works our protocol is either more secure at similar costs (see proxy-based techniques in Section 3.1) or is similar secure at better costs (see DC-net based techniques in Section 3.2).

1.3 Discussion on Significance of this Work

This section first shows the motivation of different institutions for collecting private information and the consumer's incentives for providing that information. The second part of this section gives an overview on regulations

concerning the protection of privacy and the collecting of personal information.

1.3.1 Current Situation

It is often necessary to collect personal information before a service can be provided. For example, contact and billing information is needed in order to buy a book at Amazon. But companies are also interested in collecting additional information from the customer. This information is used for market analysis and targeted advertising. In the case of Amazon, if a customer buys a product A, he later receives emails with recommendations for another product B, if there are other customers who bought both products A and B.

Many people publish private information on personal websites or in social networks, such as Facebook. Most of them are not aware that this information can be used for undesired purposes. Companies comb through the Internet to find information on job candidates. A risqué page on Facebook can have negative effects on finding a job.

Companies try to collect contact information by offering small discounts (PayBack) or free raffles. Also people often forget to make a tick on a form that says that the personal information given must not be used for marketing purposes. The main problem of giving away information is that it is hard to make sure that the information is only used as agreed, and subsequently to delete it, if desired. Old versions of websites may even still be available via services such as [1], so that deleted content of websites can still be accessed.

In other cases, the users of a service are not aware that private information is being collected. The business model of Google is based on advertising. Companies only pay for the advertising if people visit their web site over a link from Google. For that reason, Google tries to place advertising that might be interesting for the user of Google. In order to do this, Google collects information on search requests and web sites visited. Tracking of users is also used for billing reasons: the companies pay per click on the advertising.

The type of private information that may be stored and the purpose for which it may be used is regulated in privacy policies. Section 1.3.2 gives more details on this topic. But even if the user agrees to the privacy policy, he cannot be sure that the company will follow the rules. And even if the

company follows the rules, it might be possible that the information is not adequately secured, or that information is leaked by employees. This happened, for example, at Deutsche Telekom, where employees sold information on customers in 2006 [2], and also at the LGT Bank in Liechtenstein (see [3]), where leaked customer information was used to detect tax fraud. In the UK personal information on UK citizens is being lost in unprecedented rate. In 2008 millions of addresses, bank account details, and private health information was stolen or lost [4].

In IP-based communication, when users do not directly offer private information, companies or public institutions have good chance of collecting enough data from different sources to assemble the wanted information, such as websites visited and mails sent. Encrypting the communication reduces the information that can be read by third parties but information on the communication partners is still visible. Additional techniques have to be applied to hide this information. The most popular way to hide the communication link is called onion routing. Details on onion routing will be presented in Chapter 3. Successful attacks on this and other protocols show that more investigations in this area are required.

1.3.2 Law and Legal Issues

In addition to being covered by contract law as it applies to organisations' privacy policies, privacy protection and data retention is regulated in the European Union. Basically, the privacy protection law represents the interests of consumers by giving rules on how private data may be processed. In contrast, the data retention law each member of the European Union mandates the storage of connection data for all electronic communications, such as telephony and Internet communication. The following gives more details on these regulations.

Privacy

Privacy protection means the protection of personal data from misuse. In the European Union it is regulated in the convention [5]:

> The purpose of this convention is to secure in the territory of each [member state] for every individual, whatever his nationality or residence, respect for his rights and fundamental freedoms,

> and in particular his right to privacy, with regard to automatic processing of personal data relating to him ("data protection").

The convention also says that personal data may only be stored for as long as it is needed. For example, accounting information has to be deleted after a given period of time. Additional to this, the convention [5] says that personal data of an individual can be used in the interest of "State security, public safety, the monetary interests of the State or the suppression of criminal offences", in the interest of the individual or the rights and freedoms of others.

Personal data can be used for research purposes if there is no risk of infringement of privacy. In all other cases, the convention says that each individual can decide whether, by whom and when his personal data should be accessible by other parties. Privacy policies give information on what data is collected, for what reason the data is collected, how the data is secured and with whom the data may be shared. More details on privacy policies can be found in Section 2.1.4. Anonymous and unlinkable communication can help to keep personal information secret.

Data Retention

The term 'data retention' means the obligation on service providers to store information concerning electronic communications without initial suspicion. Directive 2006/24/EG [6] of the European Parliament regulates data retention within the European Union. In the directive it is stated that data on electronic communication has to be stored for the purpose of investigation, detection and prosecution of serious crime. The member states have to guarantee that this data is retained at least six months up to maximally two years. This data may be provided to national authorities only in accordance with national law.

For every communication type, the directive gives rules on which data has to be stored. For phone calls, the numbers of the calling and called party, the duration and the date of the call have to be stored. For mobile phones additionally the cell where the device is connected, the IMEI (International Mobile Equipment Identity) with which a mobile device can uniquely be identified, and for anonymous SIM-cards also the date of the date of the first

use, have to be stored. These regulations are also valid for services such as SMS (short message service) or MMS (multimedia messaging service).

For VoIP communication similar regulations exist. To identify the communication partner the IP address, name and address of the caller, the date and time of the call and the duration of the call have to be stored.

In electronic mail systems both the sending and receiving of messages has to be tracked. While sending a message the IP address and the username of the mailbox of the sender are stored. Additionally all other email addresses involved, such as the receiver email address, and the date and time of sending are stored. When receiving email information, the IP address of the receiving user and the username of the mailbox, as well as the email addresses of all involved users, the IP address of the sending mail server and the date and time when the user connects to the receiving mail server are stored.

For all other Internet communication, only the mapping of the IP address to a user identity (name and address) at any point in time has to be guaranteed. This means that no information is collected on which websites a user visited or which services, other than messaging and VoIP, he used.

The directive gives no rules on how this data has to be retained. There exist different approaches for the data retention: The British government plans to build a central database which is filled with records from Internet service providers and telecoms companies. The data will be stored for 12 months and shall be accessible only with permission from the courts. The disadvantage of this solution is that a central database would be a greater risk for attacks and abuse. In fact there have been numerous incidents, where private information was lost by governmental institutions (see for example [7]).

In Germany providers for telecommunication and Internet access are forced to store the data for at least six months and maximally seven months. But there is a considerable debate on this. Besides the problem of additional costs for storing the necessary information, it is also not clear if these companies are able to guarantee that personal information is adequately secured. The loss of personal information of 17 million customers of Deutsche Telekom in 2006 is an evidence that this is not the case [2]. In this incident data sets of customers were copied by employees of Deutsche Telekom and sold to various companies. These data sets are still on the market. As a consequence of this breach of privacy Klaus Jansen, the head of the BDK (Bund

Deutscher Kriminalbeamter), said that personal customer data should not be stored by private companies. He recommends a central data base maintained by the government and under supervision of data protection officers [8] as is planned in Great Britain.

Besides the problem that data retention is an attack on privacy and introduces the danger of misuse, it is also possible to bypass data retention. Mail servers outside the European Union do not need to store connection data of their users. Direct communication tracking of web traffic is also not possible, because only the mappings of IP addresses to their users are logged. Further investigations have to be made in order to know if someone contacted a server.

1.4 Contributions

The main parts of this thesis are published by the author. We introduced measures for unlinkability and anonymity in Chapter 2.1. The measure for anonymity was introduced by [9] and [10] while the measure for unlinkability was introduces by us in [11].

The slotted packet counting attack presented in Chapter 4 was published in [12]. It mainly improves traditional packet counting attacks by taking account to the alteration of traffic load. We showed how to apply this attack to different scenarios by optimizing the parameters of the attack.

In Chapter 5 we present a protocol for unlinkable communication. The core of the protocol and parts of the evaluation are published in [11]. Extensions for applying it to mobile ad hoc networks are published in [13, 14, 15], where [14] was awarded with the 'Best student paper award' at the Australasian Information Security Conference 2008.

Furthermore the author of this thesis was co-author of several publications in the area of trust [16, 17, 18, 19, 20, 21, 22, 23, 24, 25]. Applying these techniques to anonymous and unlinkable communication is planned.

1.5 Overview

The rest of the thesis is organized as follows: In Chapter 2 we introduce quality measures for evaluating protocols for anonymous and unlinkable communication. To do this, we define anonymity and unlinkability and introduce

measures for both, and also for performance. We also introduce the attacker model, which influences the degree of anonymity and unlinkability.

The following Chapter 3 presents existing protocols for anonymous and unlinkable communication. We divide the protocols in two categories: Proxy-based and dining cryptographers-based approaches. In the first case, messages are sent via other stations in order to confuse an observer of the communication. The anonymity and unlinkability in the dining cryptographers protocol is based on superposed sending and broadcasting messages.

Attacks on protocols for anonymous and unlinkable communication are presented in Chapter 4. The attacker either tries to infer communication links by passively observing the network traffic, or by actively manipulating the network traffic or a part of the system. Section 4.2 introduces a new passive attack, which improves traditional packet-counting attacks.

We introduce the new protocol for unlinkable communication in Chapter 5. We first introduce the protocol, followed by an analysis of performance and unlinkability. Section 5.3 shows how the protocol can be applied in different network scenarios.

The final Chapter 6 contains a critical assessment of the thesis. Section 6.2 gives directions for further studies.

Chapter 2

Quality Measures for Anonymous and Unlinkable Networks

To evaluate different systems we need a measure that expresses quality. By such a measure we can not only evaluate, but also improve and aid understanding of systems. For this purpose we introduce measures for anonymity and unlinkability that are based on the information theoretic notion of entropy. We will use these measures throughout this thesis to evaluate various anonymity systems. Also, the measure makes it possible to measure the influence of attacks on the degree of anonymity and unlinkability.

The degrees of anonymity and unlinkability depend not only on the technique applied by the user, but also on the capabilities of the attacker. Therefore, we define two different types of attacker, for use in this thesis: Passive attackers try to find communication links only by passively observing network traffic, while active attackers are additionally able to manipulate traffic and might be able to take over subsets of the users and parts of the system.

This chapter is organized as follows: In the first Section 2.1 we introduce the information theoretic notion of entropy, also known as uncertainty. We will use entropy to measure anonymity and unlinkability. The term privacy is often used synonymously to anonymity and unlinkability. We introduce the definition of privacy in order to clarify the difference. The performance measures will also be used in this thesis for evaluating protocols for unlinkable

and anonymous communication. Section 2.2 introduces the attacker model. We define two different types of attackers: passive and active attackers.

2.1 Measures

A measure in mathematics (see [26]) is defined for a σ-algebra X which is a special kind of collection of sets. It is a function μ that maps to every set $s \in X$ a real value $\mu(s) \in [0, \infty]$, where the empty set is mapped to 0. Additionally, the measure is countable additive, also called σ-additive. It can be used for quantifying different kinds of sets, e.g. lengths or volumes. We introduce measures for anonymity and unlinkability, which may then be used for evaluation. The measures give an understandable and reproducible notion for these terms. The approach to measure anonymity and unlinkability used in this thesis is based on the information theoretic notion of entropy.

2.1.1 Information Theoretic Measures

Information theory introduces the notions of *information* and *entropy*. Detailed information can be found in [27, 28]. Information theory concerns the observation of events and reasoning about other events. For example, in a communication system, the receiver tries to reason about the signal sent by observing the signal received. Therefore information is defined as a function of two variables: the observable event $y \in Y$ and the event $x \in X$ one wants to reason about, where X and Y are random variables.

Definition 1. *The information one receives concerning $x \in X$ while observing an event $y \in Y$ is denoted by $I_{X,Y}(x; y)$. It is defined by:*

$$I_{X,Y}(x; y) \stackrel{def}{=} \log_2 \frac{P_{X|Y}(x|y)}{P_X(x)}, \tag{2.1}$$

under the assumption that $P_X(x) \neq 0$ and $P_Y(y) \neq 0$. This is known as the mutual information *of x and y.*

Information is measured in bits. The measure gives the minimal number of bits used to represent the information. $I_{X,Y}(x; y)$ is symmetric in the arguments x and y. It reaches its minimum, $-\infty$, in the case of $P_{X|Y}(x|y) = 0$ and its maximum, $-\log_2 P_X(x)$, in the case of $P_{X|Y}(x|y) = 1$. A value of

0 means that the events $x \in X$ and $y \in Y$ are independent. It follows that the values for $I_{X,Y}(x;y)$ are in the following interval:

$$-\infty \leq I_{X,Y}(x;y) \leq -\log_2 P_X(x). \tag{2.2}$$

The above definition is for any two events. With the following definition for $I(X;Y)$ we can measure the mean information conveyed by two random variables.

Definition 2. *$I(X;Y)$ is defined by:*

$$I(X;Y) \stackrel{def}{=} \overline{I_{X,Y}(x;y)} = \sum_{x \in X} \sum_{y \in Y} P_{X,Y}(x,y) \log_2 \frac{P_{X|Y}(x|y)}{P_X(x)} \tag{2.3}$$

It is known as the mutual information *of X and Y.*

The *self-information* of an event $x \in X$ is the amount of information that is needed to uniquely identify it:

$$I_{X,X}(x;x) = -\log_2 P_X(x). \tag{2.4}$$

We can define the mean self information of a random variable in the same way as for the mutual information. It is denoted by $H(X)$.

Definition 3. *$H(X)$ is defined by:*

$$H(X) \stackrel{def}{=} \overline{I_{X,X}(x;x)} = -\sum_{x \in X} P_X(x) \cdot \log_2 P_X(x) \tag{2.5}$$

It is known as the entropy *or* uncertainty *of a random variable X.*

We assume that $0 \cdot \log_2 0 = 0$, so that in cases where $P_X(x) = 0$ the entropy and mutual information are well-defined. The maximum of $\log_2 n$ is reached when the random variable $X = \{x_1, \cdots, x_n\}$ has a rectangular distribution. The entropy reaches its minimum, 0, when the probability of one element $x_i \in X$ is 1 .

The entropy of a random variable X gives the maximum amount of information that can be inferred through observation. It depends only on the probabilities of the containing elements $x \in X$. Figure 2.1 shows the entropy of a rectangularly distributed random variable. The number of elements of

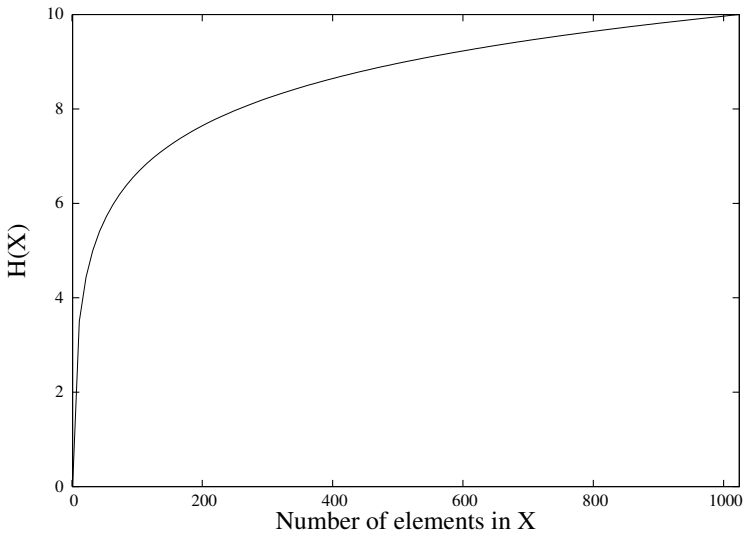

Figure 2.1: Entropy of a rectangular distributed set X

the random variable varies from 1 to 1024, while the entropy attains values from 0 to 10.

Entropy will be used throughout the thesis for the measurement of anonymity and unlinkability. Mutual information aids in the quantification of the amount of information that is revealed during different attacks on anonymity systems.

2.1.2 Anonymity

The word anonymous is derived from Greek and has the meaning 'nameless', 'of unknown name', 'wanting a name' [29]. We will use the commonly-used definition of Pfitzmann and Köhntopp [30], that is more general:

Definition 4. Anonymity *of a subject means that the subject is not identifiable within a set of subjects, the* anonymity set.

The definition makes it clear that anonymity is a property that cannot be achieved by one subject itself, in contrast to for instance confidentiality. To be anonymous, a subject needs other subjects in order to hide its identity in the corresponding anonymity set. Anonymity depends on the point of view:

If a person has enough knowledge to identify the subject of interest, then apparently this subject is not anonymous in this person's perspective. On the other hand, if there is a second person that has less knowledge making him unable to identify the same subject of interest, then, from the second person's perspective, the subject is anonymous. For example, suppose the sender of a message knows the receiver of the message but uses a protocol that makes it impossible for an outsider to detect the receiver of messages. In this case, the receiver is not anonymous to the sender but is to outsiders.

Anonymity is always related to an action. In the case of *sender anonymity* with the *sender anonymity* set it is related to sending a message. In the same way, for receiving messages, we have *receiver anonymity* with the *receiver anonymity set*. The anonymity set consists of all subjects that might have performed an action of interest. In the case of sender anonymity, the anonymity set consists of all nodes that might have sent a message of interest.

According to the definition, a subject is anonymous if it is not identifiable within a set of subjects. However, this definition makes no statement on the quality of anonymity. A subject is anonymous if the anonymity set consists of at least two members. Nevertheless, a bigger anonymity set complicates attacks on the anonymity system. This means that it is harder to reveal an acting subject with the help of statistical attacks when the corresponding anonymity set is larger. In order to evaluate anonymity systems, we introduce a measure for anonymity in the following section.

Measuring Anonymity

The measure for anonymity introduced in this section is based on an information theoretic approach. We define the *degree of anonymity* as the entropy associated with a subject. Entropy as a measure for anonymity was also proposed independently in [9] and [10]. In the case of sender anonymity, the degree of anonymity gives the number of bits that is needed to identify the sender node. By using this approach, not only does the size of the anonymity set influence the degree of anonymity, but also the probability distribution over all members of the anonymity set. As result it is possible that, even with a large anonymity set, the degree of anonymity is poor. In the example of sender anonymity, this can be the case if one sender sent the message with high probability, while the probability assigned to the rest

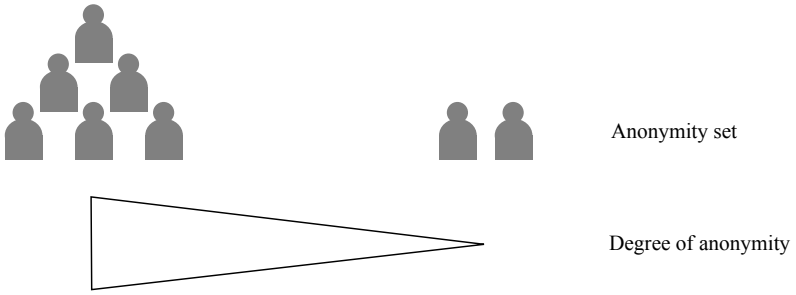

Figure 2.2: Anonymity sets and degree of anonymity

of the possible sender nodes is small. This can happen if the attacker performs traffic analysis attacks, where he observes the network traffic over an extended period of time and statistically analyses the captured traffic data. Additionally, a-priori information can influence the probabilities assigned to the members of the anonymity set.

It is assumed that the anonymity set is given by $A = \{a_1, \cdots, a_n\}$, where a_i for $0 \leq i \leq n$ are the subjects of the anonymity set. Let $E(A) = \{e(a_1), \cdots, e(a_n)\}$ the event set of A where an event $e(a_i)$ means 'a_i is the subject of interest'.

Definition 5. *The* degree of anonymity *of an anonymity set* $A = \{a_1, \cdots, a_n\}$ *is defined by:*

$$d_A \stackrel{def}{=} H(E(A)) = -\sum_{e(a_i) \in A_e} P(e(a_i)) \log_2 P(e(a_i)). \tag{2.6}$$

As seen in Section 2.1.1, the degree of anonymity is bounded by:

$$0 \leq d_A \leq \log_2 n. \tag{2.7}$$

In the case of 0, it is guaranteed that there is no anonymity: the attacker has all information to identify the subject of interest. The maximum value of $\log_2 n$ is reached when all members of the anonymity set are equally likely to be the originator of the action. Figure 2.2 shows the influence of the size of the anonymity set on the degree of anonymity under the assumption that A has a rectangular distribution.

This definition shows the two ways of improving anonymity. We can increase the degree of anonymity by increasing the size of the anonymity set,

or by equalizing the probabilities assigned to the members of the anonymity set.

The assigned probabilities may vary with the point of view. For example, if an attacker has only incomplete knowledge of the network traffic, he might assign different probabilities compared to an attacker that has absolute knowledge of the network. Thus, the degree of anonymity depends not only on the anonymity system used and its parameters, but also on the attacker. For an objective evaluation we have to define the network parameters, the capabilities of the attacker and the previous knowledge of the attacker that helps to break the anonymity. Details of the attacker model are given in Section 2.2.

2.1.3 Unlinkability

We will use a definition of unlinkability introduced by Pfitzmann Köhntopp in [30]:

Definition 6. Unlinkability *of two or more items of interest from an attacker's perspective means that the attacker cannot sufficiently distinguish whether these items of interest are related or not.*

In our work we concentrate on the term unlinkability in the context of network communication, i.e. *sender and receiver unlinkability*. That means that the attacker is not able to link the sender and the receiver of a message. In the same way as anonymity, unlinkability depends on the point of view. Two persons might make different judgments on the unlinkability of two items. Note that for unlinkability it is not necessary that the sender and receiver are anonymous. A pair of sender and receiver nodes can be unlinkable even if both the sending of a message at sender side and the receiving of a message at receiver side can be detected. This is, for example, the case in Mix systems, as we will see later in this thesis (see Section 5). On the other hand, unlinkability is necessary for anonymity. If the attacker can map a sender to the respective receiver, no sender and receiver anonymity can be guaranteed. Therefore, unlinkability is necessary but not sufficient for anonymity.

In the same way that we defined anonymity sets we can also define unlinkability sets. These sets consist of all combinations of sender and receiver nodes. For example if the attacker is able to reduce the number of possible

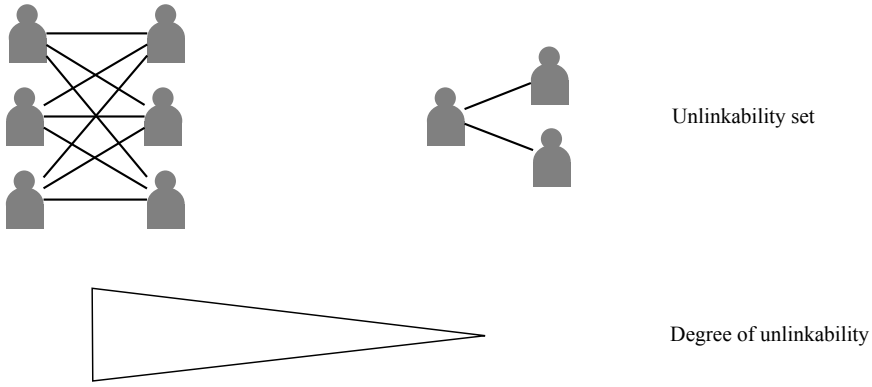

Figure 2.3: Unlinkability sets and degree of unlinkability

sending nodes to three and the number of possible receiver nodes to four, the resulting unlinkability set consists of 12 elements. Figure 2.3 shows two different unlinkability sets; the left set with nine elements and the right set with only two elements. It is harder for an attacker to detect the communication link if the unlinkability set is bigger. As with anonymity this is not the only criterion. In the following subsection we introduce a measure for unlinkability that takes the size of the unlinkability set into account.

Measuring Unlinkability

In this section we introduce a measure for evaluating unlinkability in the context of network communication. We introduced the measure in [11]. The measure is similar to that of anonymity that is based on the notion of entropy (see Section 2.1.2). We define the set of possible sender nodes as $S = \{s_1, \cdots, s_n\}$ and the set of possible receiver nodes as $R = \{r_1, \cdots, r_m\}$. We denote the communication link where s_i sent a message to r_j with the notation $l_{i,j}$. The set of all possible communication links is $L = \{l_{1,1}, \cdots, l_{n,m}\}$. Let $E(L) = \{e(l_{1,1}), \cdots, e(l_{n,m})\}$ be the event set of L where an event $l_{i,j}$ stands for 'there exists a communication link between s_i and r_j'.

Definition 7. *The* degree of unlinkability *for an unlinkability set* $L = \{l_{1,1}, \cdots, l_{n,m}\}$ *is defined by:*

$$d_L \stackrel{def}{=} H(E(L)) = -\sum_{i=1}^{n}\sum_{j=1}^{m} P(e(l_{i,j}))\log_2 P(e(l_{i,j})). \tag{2.8}$$

We see that the degree of unlinkability depends on the size of the unlinkability set and the probabilities of the possible communication links. The attacker might be able to adjust the probabilities by use of a-priori information or by use of statistical analysis of the network traffic. We can increase the degree of unlinkability by increasing the size of the unlinkability set, or by equalizing the probabilities of the possible communication links.

For a fixed size of the unlinkability set, the best case is attained if the probabilities of the communication links have a rectangular distribution. In the example of Figure 2.3, the degree of unlinkability for the right setting with only 2 links is 1, under the assumption that the links have a rectangular distribution. For the left setting, the degree increases to $\log_2 9$. In the worst case, the degree reaches its minimum of 0. This is the case when the attacker has a proof for the link between the sender and receiver node. Therefore, the bounds for the degree of unlinkability are:

$$0 \leq d_l \leq \log_2 |L|. \tag{2.9}$$

In the same manner as for anonymity, it holds that, when evaluating systems that provide unlinkability, we have to define the capabilities and knowledge of the attacker because these parameters also influence the degree of unlinkability. We provide details of the attacker model in Section 2.2.

2.1.4 Privacy

In this section we define the differences between privacy and anonymity. The terms privacy and anonymity are often used synonymously. In this thesis we distinguish between these notions. While anonymity, as we have seen in Section 2.1.2, means that a subject is not identifiable in an anonymity set, privacy means that the subject is able to reveal personal information selectively. Consequently, privacy is a weaker criterion than anonymity. Privacy implies that the subject decides which information he wants to reveal and who has access to the revealed information. In order to avoid misuse, it is useful to reveal as little information as possible. Personal information can be the basis of discrimination: some insurance companies or provider exclude persons because they belong to risk groups (see e.g. [31]). An example for privacy is that of patient confidentiality, where information on a

subject's health is revealed. Access to this health information is limited to the physicians involved.

Privacy is usually enforced by privacy policies. Privacy policies are rules which are introduced by organizations in order to ensure the privacy of their customers. Such policies focus on personal data. They give information on:

- How the data is collected
- What data is collected
- Intended use of the data
- Under which conditions the data is passed to third parties
- Application of the policy and procedure for handling complaints.

These policies are published, for example on the web site of the organization. As a case in point, in the European Union, web site providers have to inform their users on privacy issues. The problem of privacy policies is that it is hard to verify whether the policies are fulfilled. In the end, the user has to trust that the organization follows the policies. This is the main difference from anonymity and unlinkability. If a user communicates in an anonymous or unlinkable manner, the organizations involved are not able to obtain any personal information. Therefore, no trust in these organizations is required.

2.1.5 Performance

In order to use a protocol for unlinkable or anonymous communication in practice, it must not be efficiently breakable. On the other hand, the performance of the protocol has to be acceptable. For measuring the performance of communication protocols in this thesis we use the throughput together with the related message overhead and the delay of messages.

The throughput is the average rate of successfully delivered messages. It is measured in bits per second. In the case where the packet length is constant it can be measured in packets per time interval.

The delay during a communication is the average time needed to send a packet from the source to the destination, i.e. the time between sending a message at the source node S and receiving this message at the destination node D.

In this thesis we measure the throughput and the delay in absolute values with the help of a prototype implementation of the presented protocol. We evaluate the related work by comparing the throughput and delay of the network and the communication link while using the protocol of interest, compared to the delay and throughput of the network without using the protocol.

2.2 Attacker Model

It is important for the evaluation of anonymity systems to define the capabilities of the attacker. With one attacker model, a system might be secure, while the same system with the same parameters is insecure when assuming a different attacker model. The attacker model gives information on the capabilities of an attacker and additional knowledge. In general, we assume, as in [32] (Shannon's maxim) and in [33] (Kerckhoffs' principle), that the attacker knows the system that is used for providing anonymity, and unlinkability respectively. This includes the knowledge of the algorithms used, e.g. for encryption, but not the private keys of the members of the network. In our attacker model, the computational power and storage is bounded, so that the attacker is not able to break cryptographic functions. There are two basic attacker models that differ in their capabilities: passive attackers and active attackers. The following subsection will discuss these basic models in more detail. For the evaluation of anonymity and unlinkability systems, this thesis assumes the strong passive attacker model. The impact of active attacks on protocols for anonymous and unlinkable communication will also be discussed.

2.2.1 Passive Attacker

In this thesis we assume the strongest passive attacker that is defined as follows.

Definition 8. *A passive attacker is able to observe every sending and receiving event in the network and he is able to see the content of the all packets sent and received.*

The setup of an passive attack is shown in Figure 2.4. There are two main reasons for this approach: First, it is easier to compare and evaluate the

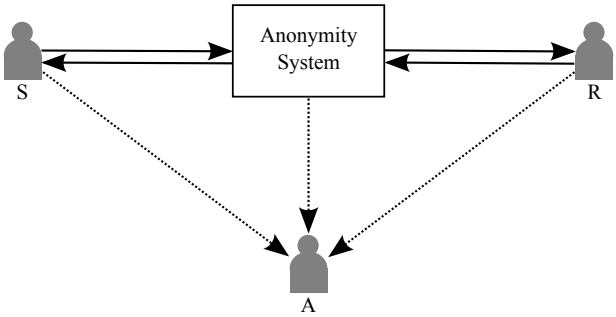

Figure 2.4: Passive attacker of an anonymity system

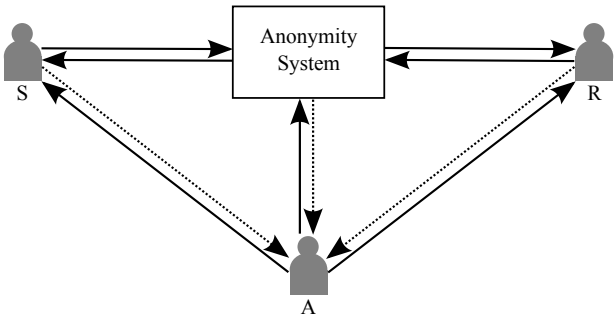

Figure 2.5: Active attacker of an anonymity system

results of different systems. Second, if a system successfully works under the assumption of this attacker model, it will also work with a weaker attacker model. This attacker model is often used in literature (e.g. [34]). These capabilities are sufficient to perform traffic analysis attacks. In these attacks the attacker tries to perform statistical analysis on captured traffic in order to reveal the communicating nodes. Section 4.1.1 shows this kind of attack in more detail. This attacker model will be used for evaluating the proposed protocol for unlinkable communication.

2.2.2 Active Attacker

In addition to the capabilities of a passive attacker, an active attacker has the following capabilities.

Definition 9. *Additionally to the capabilities of a passive attacker an active attacker is able to actively manipulate messages or the system. An attacker who is able to inject, alter, and block messages is known as Dolev-Yao at-*

tacker [35, 36]. Additionally, an active attacker might be able to take over some of the system's users and also parts of the system itself.

The setup of an active attack is shown in Figure 2.5. The effectiveness of such attacks is increased by the collaboration of several active attackers. In most cases the attacker uses active manipulations to cause situations in which passive attacks are more powerful. In Section 4.1.2 of this thesis, active attacks on anonymity systems are introduced. We will show the influence of active attackers on the protocol proposed in Section 5.

2.3 Conclusion

In this chapter we introduced anonymity, unlinkability and performance measures. The degrees of anonymity and unlinkability depend not only on the protocols, which are used to guarantee anonymity and unlinkability, but also on the capabilities of the attacker. Therefore, we also defined the capabilities of the attacker in this chapter. We introduced two different kinds of attacker: Passive attackers are able only to observe the network traffic. From the captured information, the attacker can reason about communication links. Active attackers are additionally able to manipulate traffic and might be able to take over parts of the system and subsets of the users.

For the evaluation of the protocol presented for unlinkable communication (see Chapter 5) we will assume a global passive attacker, i.e. an attacker that is able to observe every communication in the network. Using the information theoretic notion of entropy we can show that, during the execution of the protocol, no information on the communication links is revealed.

A global passive attacker is also assumed for the slotted packet counting attack, which is presented in Section 4.2. Attackers that have only incomplete knowledge of the network traffic also have a good chance of revealing communication links. In Chapter 4 we present another attack, where the attacker actively manipulates traffic. A table at the end of Chapter 4 shows the influence of attacks on the degree of anonymity and unlinkability while using the protocols introduced in Chapter 3.

Chapter 3

Anonymous and Unlinkable Networks - State of the Art

Research in the area of unlinkable and anonymous communication started in 1981 with the concept of Mixes by Chaum [37]. Detailed information will follow in Section 3.1.2 of this chapter. This approach was only considered for the use in electronic mail systems. In this setting, Mixes are able to offer unlinkability and, as long as all other users of the Mix and the Mix itself are trustworthy, the attacker has no chance of breaking the unlinkability. This technique was later applied to more general communication settings. However, in these settings it cannot be guaranteed that the attacker infers no information from passive observation of the incoming and outgoing links of a Mix. A second drawback of traditional Mixes is that they introduce some delay in forwarding packets. This small delay does not matter for sending mails. In other settings, such as web browsing, these delays are not tolerable. The basic idea of Mixes is adapted in a number of manners to overcome the problem with the delays. Among these, the most popular technique is Tor (see Section 3.1.3), which may be described as a low-latency Mix. In essence, the mixing step is removed in order to reduce delays.

A second means of achieving unlinkability and anonymity is the dining cryptographers' network, also introduced by Chaum [38]. In this technique, the users build groups in which they can send and receive messages anonymously. Sender anonymity is guaranteed by superposed sending, and receiver anonymity by broadcasting all messages within the group. In theory, this technique offers unbreakable unlinkability and also sender and receiver ano-

nymity. However, there are several weak points in the protocol, which can be exploited by an active attacker. The second drawback of this approach is its poor scalability. Because of the superposed sending and the broadcasts, the message overhead increases quadratically with the size of the anonymity set. This illustrates the basic problem of most techniques: increased security comes at the cost of performance and vice versa.

The remainder of this chapter is organized as follows: Section 3.1 introduces proxy-based approaches for the implementation of unlinkable and anonymous communication. These approaches have in common that the traffic is channeled through other stations before it reaches the receiver, see e.g. Mixes. The second approach will be presented in the following Section 3.2, which is based on the dining cryptographers' network.

3.1 Proxy Based

The idea of the first systems which were built to offer unlinkability, was to route traffic over intermediate nodes, such as proxy servers. The common weakness of these systems is that a strong passive attacker, as assumed in Section 2.2, is able to determine the sender and the receiver of a communication. Due to this fact, he is able to perform statistical analysis on the network traffic, as we will see in Chapter 4. In the following, we will describe the evolution of proxy-based systems, starting with a simple proxy server and ending up with Mix networks and onion routing.

3.1.1 Proxies

A *proxy server* is a device in the network that passes the requests of its users to their destinations (see [39]). Thus, if a user A wants to contact B, it sends its request to the proxy server P and P forwards the request to B. All messages from B to A are also passed over the proxy (see Figure 3.1). As result, node B has no direct contact with A and therefore does not need to know the identity of A. The only contact address for B is the address of the proxy server P. Only P and A know the end-points of the communication.

This simple solution is able to hide the sending node from the receiving node, and the traffic overhead is small. On the other hand, the users have to trust the proxy, because it is a single point of failure. Failure affects not only service availability, which can be broken by heavy load or by de-

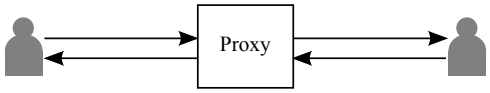

Figure 3.1: Using proxy server for unlinkable communication

nial of service attacks, but also security. A proxy knows the communication links and is therefore able to reveal this information to third parties. To weaken this threat, the user can build paths over several proxies. In this case the communication links can only be revealed if all proxy servers on the path cooperate in a malicious manner. Such an increase in security comes at the cost of performance: due to longer communication paths, the delay while sending messages and the amount of packets sent increases. Even with longer communication paths, the sending and receiving nodes of a communication are still visible to a strong passive attacker. The attacker can use this information to perform statistical attacks.

The throughput while using proxies depends mainly on the bandwidth of the proxies. When several users use one proxy, they have to share its bandwidth, with a consequent decrease in the throughput of every communication. The delay of a proxy depends mainly on its computational power. In the best case, the delay linearly increases with the number of proxies on the communication path.

3.1.2 Mix Networks

Mixes were introduced by Chaum [37] in order to offer untraceable electronic mail. To send a message to a destination node using a Mix, the sender first has to encrypt the message together with the address of the destination node. In [37] Chaum proposes the use of asymmetric encryption, but solutions with more efficient symmetric encryption are also possible. In a *Mix round*, the Mix server waits until it has received n encrypted messages, where n is fixed. It decrypts these messages and sends the messages reordered, e.g. sorted lexicographically, to the destination nodes. In the case of asymmetric encryption, additional random bits must be added to the encrypted message. If the messages sent to the Mix do not contain random bits, the attacker is able to link the incoming to the outgoing traffic by re-encrypting forwarded messages with the public key of the Mix. The view of a passive attacker is shown in Figure 3.2.

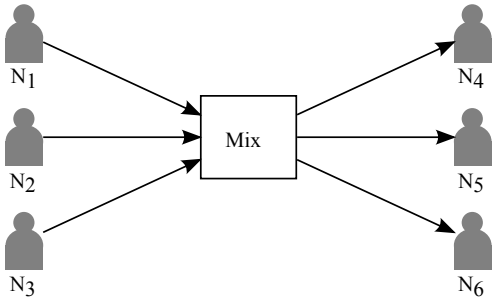

Figure 3.2: Example for Mix server

For the return communication, Chaum proposes the use of untraceable return addresses, which are constructed by the source node and sent to the destination node.

In the following, we show by example the functionality of a Mix. Figure 3.2 shows a Mix server with inputs from nodes N_1, N_2, N_3 and outputs to N_4, N_5, N_6. Assume, that the Mix owns a pair of public and secret keys PK_{Mix} and SK_{Mix}. N_1 wants to send message M_1 to N_5, N_2 wants to send M_2 to N_4 and N_3 wants to send M_3 to N_6. To do this, N_1, N_2 and N_3 build the following messages and send them to the Mix:

$$\begin{aligned} N_1 \rightarrow Mix: &\quad [M_1, Random_1]_{PK_{Mix}} \\ N_2 \rightarrow Mix: &\quad [M_2, Random_2]_{PK_{Mix}} \\ N_2 \rightarrow Mix: &\quad [M_3, Random_3]_{PK_{Mix}} \end{aligned}$$

The Mix server receives the messages and decrypts them with its private key SK_{Mix}. Afterwards, it sends the following messages in lexicographical order to the receiver nodes:

$$\begin{aligned} Mix \rightarrow N_4: &\quad M_2 \\ Mix \rightarrow N_5: &\quad M_1 \\ Mix \rightarrow N_6: &\quad M_3 \end{aligned}$$

If N_1 wants to receive a reply from N_5 it can generate an untraceable return address by choosing a one-time key $Random_4$, which also serves as a random string, its address A_1 and a newly generated public key PK_1. The untraceable return address is as follows:

$$[A_1, Random_4]_{PK_{Mix}}, PK_1$$

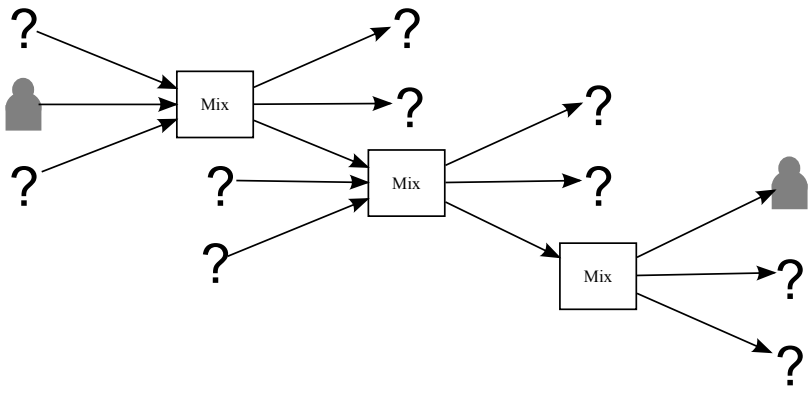

Figure 3.3: Cascade of Mixes

N_1 sends this return address as a message over the Mix to the receiver N_5. N_5 also generates a random string $Random_5$ and sends its reply back to N_1 as follows:

$$\begin{aligned} N_5 \rightarrow Mix: &\quad [A_1, Random_4]_{PK_{Mix}}, [M_5, Random_5]_{PK_1} \\ Mix \rightarrow N_1: &\quad A_1, [[M_5, Random_5]_{PK_1}]_{Random_4} \end{aligned}$$

It is also possible to build cascades of Mixes (see Figure 3.3). Sending and receiving of messages work in the same way, as with a single Mix. The necessary encryption is applied recursively for every Mix on the communication path.

In order to avoid the easy revelation of communication links by message replay, the Mix has to check that none of the incoming messages has been already processed. The encryption of a replayed message would be the same as that of the original message. By intersecting outgoing links of two Mix rounds in which the same messages appear, the attacker can link the source and the destination.

Because this kind of Mix has to wait until it received n messages in a round, the delay becomes arbitrarily long in the worst case. On the other hand, the unlinkability set for such a Mix consists of n^2 elements. This set increases exponentially by adding more Mixes to the communication path. For a path that contains two Mixes, the unlinkability set increases to n^4. If the attacker is able only to observe one Mix round and does not control any of the Mix users and the Mixes, he is not able to make any statement concerning the communication links. However, by observing Mixes over a longer period

of time, an attacker might be able to probabilistically reveal communication links through statistical analysis. Adding dummy traffic reduces the impact of statistical attacks, but reduces the efficiency of Mixes.

Some years after Chaum's initial proposal, new types of Mix were proposed. In order to provide anonymous real-time communication, Jerichow et al. [40] proposed real-time Mixes, which extend the functionality of traditional Mixes.

A timed Mix [41] sends the collected messages periodically, once fixed waiting time t has elapsed. This overcomes the problem with delays, but has the drawback that in the worst case the unlinkability set consists of only one communication link. These timed Mixes can be combined with traditional Mixes in two ways: in the first case, the Mix sends all collected messages at once when a limit of n messages is reached or a fixed waiting time t has expired. In the second case, the Mix sends all collected messages at once when the message limit n is reached and the waiting time t is over. Several variations of these timed Mixes exist.

Stop-and-Go-Mixes provide probabilistic unlinkability. This kind of Mix forwards a message with an exponentially distributed delay. The parameter of the exponential distribution has to be chosen in such that, with high probability, the Mix forwards each message after a new one arrived. Because of the exponentially distributed delay the attacker is not able to detect the order in which the Mix forwards the messages to their destinations.

In summary, Mixes are able to offer unlinkability. The size of the unlinkability set and the delay depends on the kind of Mix and the number of users it has. In theory, a high number of users offers smaller delays at the Mix, while in practice the users have to share the bandwidth of the Mix. The delays increase linearly with the number of Mixes in the Mix cascade, while the absolute value of the delay also depends on the Mix type, the settings of the Mixes and the number of users. Even if the number of users of a Mix is high, an attacker is able to perform statistical attacks because the sender and receiver nodes are detectable. Stop-and-Go Mixes complicate such attacks. In this case, end-to-end mapping of packets is harder because of the time delay. However, statistical analysis is possible even in this case.

The Mix technique is used for example in Mixminion [42], which is an anonymous remailer protocol. JAP (Java Anonymous Proxy) [43] uses cascades of Mixes for general Internet communication.

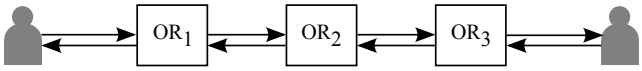

Figure 3.4: Onion routing with 3 onion router

3.1.3 Onion Routing

The concept of *Onion Routing* was introduced in [44, 45, 46, 47] and later improved in [48] under the name of *Tor*, the 'second generation onion router'. It was built for low-latency Internet communication such as web surfing. It offers bidirectional communication with a performance near to that of real-time Mixes. In the same way as for common proxies, the unlinkability of onion routing is achieved by redirecting the network communication over intermediate nodes, known as *onion routers*. To send messages, a user of onion routing first builds up a *circuit*, which is a path over several onion routers. Each of the onion routers knows only its predecessor and successor node on the path, so that a single malicious onion router is not able to link the source and the destination of a communication. While building the circuit, the source node negotiates secret keys with all onion routers on the path. For the key exchange, the source node builds a message, called an *onion*, that contains the secret keys of these onion routers. This onion is recursively encrypted with the public keys of the onion routers, with each layer of encryption containing a secret key. If an onion router receives such an onion, it is able to unwrap one layer of encryption. The encryption contains the secret key which will be used during the onward communication. The rest of the onion is forwarded to the next node on the path. In the communication phase, the source node builds onions, which consist of messages, recursively encrypted with the negotiated keys. These onions are sent over the path of onion routers. The last onion on the path obtains the messages in plaintext by decrypting the received onions and forwards these messages to the destination node (see Figure 3.4). Thus, the destination node only knows the identity of the last onion router on the path. For return communication, the destination node sends its messages to the last onion router. Every onion router on the path encrypts the onion with its secret key and passes the message back along the path to the source node.

The message overhead of the onion routing protocol increases linearly with the number of onion routers on a circuit. Computationally expensive

asymmetric encryption is used only during the build-up of a circuit. During the communication phase, efficient symmetric encryption can be used. Consequently, the delays at every onion router are small. The overall delay increases linearly with the number of onion routers on the communication path, because every onion router performs the same operations. The throughput depends on the throughput of every onion router. It cannot be better than that of the onion router with the worst throughput. With the current implementation of *Tor*, where onion routing is used, the delay and throughput is, in most cases, sufficient for web surfing.

The good performance and the low latency of onion routing come at the cost of less dependable unlinkability. Admittedly, the chance that malicious onion routers completely control a communication link decreases with bigger circuits, but nevertheless a small number of onion routers under the control of an attacker provide a good chance of revealing the source and destination of a communication. Indeed, there have been successful attacks on Tor. By exploiting some weak points of the onion routing implementation, statistical attacks can be performed while controlling only a small number of onion routers. For example, Bauer et al. [49] exploit the mechanism of Tor's optimization that favours onion routers with high bandwidth and high uptime. The implementation of Tor also contains a mechanism to periodically build a new circuit for communication. Generally, a circuit may only be used for one minute before a new one must be built. But instead of complicating attacks, since only a limited number of requests to a destination can be linked to an exit onion router, an attacker controlling only a small number of onion routers has a good chance of identifying many end-to-end links. In [50] Wright et al. introduce the *predecessor attack*, which exploits this mechanism of replacing the path.

Under the assumption of a strong passive attacker (see Section 2.2), onion routing is not able to guarantee unlinkability. Even though the onions change at every hop on the way to the destination, the attacker can easily perform statistical analysis on the traffic because the source and the destination of a communication are detectable.

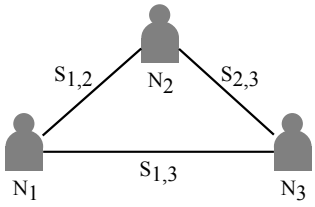

Figure 3.5: DC-net group with shared keys

3.2 Dining Cryptographers-Based Solutions

With the dining cryptographers' network (*DC-net*) [38], Chaum introduced a protocol for anonymous and unlinkable communication. The users of DC-net create groups which are used to send and receive messages anonymously. Sender anonymity is based on the principle of superposed sending; receiver anonymity on broadcasting messages.

One prerequisite for the protocol is that all nodes share pairwise secret key bits, which are only used once. The functionality of the protocol is as follows: Sending of messages is round-based, e.g. only one node is allowed to send a message in a predefined round. We will show different reservation techniques later in this section.

In the first step, the participants use an anonymous method to determine which one of them is allowed to send a single-bit message in the next round. We denote the sending node with N_1 and the other nodes of the DC-net group with $N_2 \cdots N_n$, where n is the number of nodes that participate in the DC-net group. Each node N_i, $2 \leq i \leq n$, builds the sum modulo two of all secret key bits and broadcasts this value to all members of the group. Node N_1 builds the sum modulo two of the secret key bits and the message and publishes this value. The sum modulo two of all published values results in the message that was sent by N_1.

We illustrate the protocol with an example. The DC-net group shown in Figure 3.5 consists of only three nodes N_1, N_2 and N_3 where N_1 and N_2 share the secret key bit $S_{1,2}$, N_1 and N_3 the bit $S_{1,3}$, N_2 and N_3 the bit $S_{2,3}$.

N_1 is the node that is allowed to send its message bit M in the next round. Now, each node N_i calculates the bit P_i $(1 \leq i \leq 3)$ as follows:

$$\begin{aligned} N_1: \quad P_1 &= S_{1,2} \oplus S_{1,3} \oplus M \\ N_2: \quad P_2 &= S_{1,2} \oplus S_{2,3} \\ N_3: \quad P_3 &= S_{1,3} \oplus S_{2,3} \end{aligned}$$

These values P_1, P_2 and P_3 are published, so that all nodes are able to calculate the sum modulo two of these bits. This results in the message bit M:

$$\begin{aligned} N_1, N_2, N_3: \quad & P_1 \oplus P_2 \oplus P_3 \\ &= S_{1,2} \oplus S_{1,3} \oplus S_{1,2} \oplus S_{2,3} \oplus S_{1,3} \oplus S_{2,3} \oplus M \\ &= M \end{aligned}$$

The use of this DC-net protocol, prevents an attacker from inferring any information concerning the communication link. However, there are some problems that we have not yet discussed. One problem of the protocol is that the correct delivery of broadcast messages has to be guaranteed which is known as the reliable broadcast assumption. As stated above, the secret key bits are only used once. Chaum proposes for the key distribution either the use of one-time-pads, e.g. supply disks carrying key bits to the users of DC-net, or of cryptographic pseudo-random sequence generators, where only the seed for generating the pseudo-random key string has to be exchanged. These seeds can securely be exchanged by using asymmetric encryption.

We also did not go into the details of communication rounds reservation. Chaum proposes two techniques: in the first, a node chooses a time slot in which it wants to send its messages. If there is a collision, it tries to resend the message after a randomly chosen delay. The distribution of the waiting time can be adjusted by reference to the collision rate. The second way of reserving communication rounds is by first sending a bit vector, where each bit represents a communication slot. Initially, all elements of the vector are set to 0. Each node that wants to send a message chooses one communication slot and marks it with a 1 in the vector. The values of all nodes are published at once using the superposed sending protocol above. The node that set the i-th bit of the vector is allowed to send in the i-th communication slot. With this approach, if an even number of nodes want to send at the same time

a collision of communication rounds can be detected with high probability before the sending phase of the DC-net protocol. Later, further reservation techniques were proposed: Bos and Boer [51] describe an approach based on factorization of polynomials; Pfitzmann [52] and Waidner [53] use the reservation map technique; reservation based on superposed sending was introduced by Waidner in [53]; and Mix networks were used by Studholme and Blake in [54]. Further improvements on the protocol have been made: Waidner and Pfitzmann remove the reliable broadcast assumption in [55] through the use of a Byzantine agreement protocol. The Byzantine agreement protocol gives a way how to reach a consensus between distributed parties if some of them give incorrect inputs.

The reservation phase and the superposed sending are weak points of the protocol. In Chapter 4, attacks on DC-net exploits these weaknesses to allow the attacker to disrupt the execution of the protocol, or to reveal the sender of a packet.

Besides these weaknesses the message overhead of the DC-net approach is also critical. Because of the superposed sending, all nodes have to send a message to all participants of the DC-net whenever any one node wants to send a message. By enlarging the DC-net group linearly in order to increase the degree of anonymity and unlinkability, the message overhead is increased quadratically. The delay also increases significantly because of the huge amount of messages that must be sent: every node sends and receives $n - 1$ messages during communication. The reservation phase induces additional delays. To avoid this overhead some protocols, such as Acimn [14] and Herbivore [56], limit the size of the DC-net groups.

Chaum proposes the reduction of message overhead by building cycles. Instead of broadcasting the messages the users build a ring, wherein message bits are sent only between neighboring users. To send a message, the bit has to pass around the ring twice: in the first round the message is constructed by building the sum modulo two of the message with the secret key. In the second round the message is published to all users of the ring. In this case, $2 \cdot n$ messages are needed to send only one message; additional messages are needed for the key exchange. Disrupting the protocol in this communication setting is even simpler, because the outputs are only visible to the direct neighbors in the ring. Moreover, all nodes still have to share pairwise secret keys, which are used only once.

3.3 Conclusion

The proxy-based approaches presented in this chapter do not offer anonymity but unlinkability, from the perspective of a strong passive attacker. The reason for that is that the attacker is able to observe the endpoints of a communication and therefore knows who sent packets and who received packets. Only the DC-net approach can guarantee sender and receiver anonymity, because here the attacker is not able to detect which nodes sent or received packets. In fact, it can be proven that in theory no information on the sender and receiver of a message can be gained simply by passively observing the network traffic. However, for the correct execution of the protocol, some difficult assumptions, such as the reliable broadcast assumption, must be guaranteed.

The techniques presented show that there is a connection between effectiveness and performance: if a protocol is more efficient, it is more likely that an attacker can break the unlinkability or anonymity. In general, the proxy-based approaches are efficient enough with respect to throughput and communication delay for practical use. For this reason, all protocols that are used in practice are based on this technique. On the other hand, the DC-net approach lacks scalability, which is the main reason why it is not used in practice.

In this thesis we do not present all existing protocols for anonymous or unlinkable communication. This is because most approaches are variations of the techniques presented. In the area of anonymous ad hoc communication, protocol design concentrates on route discovery, which is not the subject of this thesis.

In the following chapter we show attacks on protocols for anonymous and unlinkable communication. Most of the attacks affect the proxy-based approaches, but there exist also attacks on DC-nets. We will show how attacks can break the unlinkability and anonymity of the protocols.

Chapter 4

Attacks on Anonymity Protocols

This chapter builds on [12] and shows a variety of attacks on anonymity protocols. Exposing weaknesses in existing protocols and analyzing how attackers are able to infer information on communication links, is helpful in developing new and better protocols for anonymous or unlinkable communication. Also, these attacks show which protocols are unable to guarantee anonymity or unlinkability, and therefore should not be used in the future. From the attacker's perspective, it is interesting to know how these systems can be broken, in order to reveal communication links. With this knowledge, attacks can be performed not only by public institutions, but also by private companies for observing employees and for commercial issues.

The manner in which these attacks may be performed depends on the capabilities of the attacker; the attacker models introduced in Section 2.2 are also relevant to this chapter. Passive attackers are attackers that collect information by passively observing network traffic and making a statistical analysis of this data. The attacks in Section 4.1.1 and Section 4.2 assume this attacker model.

An active attacker is additionally able to manipulate the network traffic and might be able to control parts of the system or some of the users. We show attacks that are based on this attacker model in Section 4.1.2. These active attacks can be used in combination with passive attacks. In this case, the attacker tries to modify the system, in order that passive attacks may be performed more easily.

The attack analysis in this chapter is intended to be independent of the protocol used. In figures we will use the notation anonymous system (AS), as in Figure 4.1, to denote the infrastructure that is used for the execution of the protocol. We designate the sender by S, the receiver by R and the attacker by A.

The rest of this chapter is organized as follows: In the first part (see Section 4.1) we show a number of attacks that have been presented in the literature and categorize these attacks as passive or active. Section 4.2 introduces the slotted packet-counting attack. This attacks uses the idea of common packet-counting attacks as its starting point, i.e. the attacker counts the number of packets sent and received at two different positions in the network, but eliminates the large number of false positives by taking account for the alteration of the traffic load over time. Finally, we show the influence of the attacks presented on a variety of anonymity systems in Section 4.3.

4.1 State of the Art

In this section we give an overview of known attacks on anonymity and unlinkability protocols. We divide the attacks into passive attacks (see Section 4.1.1) and active attacks (see Section 4.1.2). Section 4.3 shows the influence of the attacks presented in this section, and in the following Section 4.2, on the protocols presented in Chapter 3.

4.1.1 Passive Attacks

Passive attacks on anonymity systems are mainly based on statistical analysis of network traffic. For this, we assume the strong passive attacker model introduced in Section 2.2.1, Defintion 8. In addition, most of the attacks can also be performed by weaker attackers, having incomplete knowledge of network traffic.

Packet-counting

In packet-counting attacks, the attacker observes traffic at two different points in the network. By counting the packets sent and received at these two points, the attacker tries to detect whether these packets belong to a single communication link. If the packet count at these two points is similar,

the attacker assumes that they belong to a single communication link. The problem of this assumption is that other communications may overlap at one of these points. This results in different packet counts even if a particular link between a sender and receiver was actually routed through these points. On the other hand, it might also be possible that there are two stations in the network that have a similar volume of sent and received packets even though the sent packets are not related to those received. In fact, by knowing only the packet counts, the attacker is able only to determine the relative frequencies of packet counts on sender and receiver side, which increase if the number of packets counted on sender or receiver side increases. In Section 4.2 we give more details on packet-counting and present an improved version.

Message Coding and Packet Volume Attacks

By knowing the content or the coding of a packet, the attacker can acquire information that helps him by linking multiple packets to a single communication. This kind of attack is known as message coding attack. For example, if the content of the packet is the same when it enters as when it leaves an anonymity system, as is the case while using proxies, the attacker can easily link these packets to a single communication. This weakness also exists in some other protocols for anonymous communication, such as in ANODR [57].

If the content of the packet is in plaintext, the attacker might receive information on the source or the destination of the packet. To avoid such attacks, packets should be not readable by the attacker and if a packet is observable at different points by the attacker, its appearance should change.

Packet volume attacks (see [58, 45]) are similar to the message coding attack with the difference that the attacker tries to identify packets belonging to a single communication link by comparing their length. To avoid this attack, most protocols for anonymous and unlinkable communication use a constant packet length. If the size of the data is smaller than the pre-defined size of the message, the sender pads the message with dummy bits until it reaches the desired size.

Timing Attacks

Timing attacks, as presented in [45, 59], use the timing of packets in order to map them to a communication link. In order to perform this mapping, the attacker estimates the latencies of packets when sent over different routes. He stores these values for use during the attack. If the attacker now wants to find the communication partner D of a node S, he observes traffic at two points: The packets that are sent from node S and the packets that are received by a suspected node P. If the latency of sending at S and receiving at P matches the pre-computed value, it is very likely that the attacker has found the destination node $D = P$.

If the attacker is only able to observe the traffic at node S, he can measure the latency between sending a message from S and receiving the reply from D. This technique can also be used for geolocation in the Internet [60]. Here, an attacker measures the latency of a ping message that he sent to a node in the Internet.

Communication Pattern Attack

The idea of the communication pattern attack is similar to that of message coding and packet volume attacks. The difference is that the attacker tries to find special features in the communication pattern rather than in the messages. As in [59], the attacker can compare the communication patterns at two different points in the network. If they are similar for some period of time, it is very likely that these communication patterns belong to a single communication link. These traffic patterns are detectable even if the communication is encrypted, overcoming a weakness of the coding and volume attacks.

Another way to use traffic patterns to reveal communicating nodes is by using the technique presented by Dusi et al. [61]. They show how a means of detecting that a blocked protocol, e.g. peer-to-peer communication, is being tunneled over a legitimate protocol such as HTTP. This is possible even if the communication is encrypted. If the attacker of an anonymity or unlinkability protocol is able to detect when a special protocol is being used for communication, he might be able to reduce the set of possible receiver nodes.

Intersection Attack

In the intersection attack [59], it is assumed that the participants in the network usually communicate with the same communication partner for some time. If the attacker observes the network over an extended period of time, he might be able to detect the time points where the target node S sends messages and other nodes receive messages. By intersecting the sets of receiving nodes, the attacker tries to reduce the receiver anonymity set.

Under the name of *disclosure attack*, Agrawal and Kesdogan [62] give a more detailed and improved version of this attack. Further details can be found in [63, 64]. In the first step of the attack, the attacker has to estimate the number m of communication partners of the target node, S, by using observations. The attacker then builds recipient sets $B_t = \{{b_t}^1, \cdots, {b_t}^n\}$ that contain all nodes that received a message from S at time point t. t is increased by one if S sends a message, e.g. at time point 1, S sends the first message.

After this, a *learning phase* and an *exclusion phase* must be carried out. In the learning phase, the attacker chooses those recipient sets B_i, such that $B = \{B_{j_1}, \cdots, B_{j_m}\}$, with $B_{j_x} \cap B_{j_y} = \emptyset$ for $x \neq y$. The elements of B are called the basis sets. Because the attacker builds sets only when S is sending, there is exactly one communication partner in the sets $B_{j_i}, 1 \leq i \leq m$.

In the exclusion phase, the attacker makes new observations which lead to a new recipient set R. This recipient set R is intersected with the basis sets. Because there is exactly one communication partner in every basis set there is at least one intersection between the new recipient set and a basis set that is not empty. Empty intersections are not possible, because Agrawal and Kesdogan assume that the attacker knows the number of communication partners m of S. If there is more than one non-empty intersection, the new recipient set is discarded. Alternatively, if there is an intersection with only one basis set B_{j_i}, the communication partner must be in the intersection of the recipient set R and the basis set B_{j_i}. A new basis set, B'_{j_i}, is built with $B'_{j_i} = B_{j_i} \cap R$. This process is repeated until every basis set contains only one node, which is a communication partner of node S.

Further work on the disclosure attack as applied to Mix networks was done by Agrawal et al. [65] and Danezis et al. [66]. Agrawal et al. show the NP-completeness of their disclosure attack and give estimates for the learning and exclusion phase. Danezis et al. improve the disclosure attack through

the use of statistical methods, so that its complexity is in P. Kesdogan and Pimenidis [67] introduce the same attack as the *hitting set attack*. They too use statistical methods to avoid a runtime complexity in NP.

The drawback of this kind of attack is that a large set of observations is needed to satisfy the assumption that the law of large numbers applies.

4.1.2 Active Attacks

In active attacks, the attacker tries actively to bring an anonymity system into a state in which he can make better conclusions concerning the communication links. The active attacker model is introduced in Section 2.2.2, Definition 9. In most cases, active attacks are used in combination with passive attacks, as described in Section 4.1.1.

Denial of Service Attack

The aim of a denial of service attack (DoS) is to make a particular resource unavailable. To do this, an attacker attempts to prevent intended users of the resource from using it. This can be done by overloading the target with dummy requests (consumption of computational resources, i.e. bandwidth, disk space, CPU time), by preventing the user from using the target (e.g. disruption of configuration or routing information), or by disrupting the network traffic by flooding the network (disruption of physical network components or communication media, e.g. jamming in wireless networks). If this attack is performed by a coordinated group of attackers, it is called a distributed denial of service (dDoS).

Such an attack can be used to prevent users from using an anonymous routing protocol. For example, an attacker might overload the Mixes of a Mix network, so that the legitimate users can no longer use the service. Additionally this attack can be used to execute more complex attacks. For example, Borisov et al. [68] apply selective DoS attacks against a variety of anonymity systems in order to break their anonymity. Later in this section we will show more attacks that make use of DoS.

Replay Attack

In a replay attack, the attacker records packets sent over the network in order to replay them at a later time. Such an attack can be used to bypass

access control mechanisms, allowing the attacker to access secured resources. For example, an attacker can record the login process to a computer in the network. Later, he can replay this login information consisting of login name and encrypted password to gain access to the computer. To gain access to secured resources, the attacker does not necessarily need to break the encryption of passwords. Such an attack can be prevented with the use of one time passwords or session keys, as in TLS/SSL [69].

Login is not the only aspect of anonymous and unlinkable communications systems that is vulnerable to replay attacks: the replay of ordinary data packets can also help to break anonymity or unlinkability. The more packets are sent over the communication path, the more accurate traffic analysis becomes. If an attacker wants to know the destination of only one specific packet, he records it and resends it several times. This technique allows the attacker to gather enough information for traffic analysis without having to wait until the sending node has transmitted a sufficient number of packets. In order to prevent such an attack, a Mix, as proposed by Chaum [37], checks every received packet to see whether it has been sent before.

Message Delaying Attack

Message delaying (see [59]) can be used to facilitate other attacks. An attacker delays packets in order to bring the network into a state where he can easily perform other attacks. The attacker can use this technique to attack stop-and-go Mixes (see Section 3.1.2). To do this, he delays packets such that only one message is separated in the Mix. It is also possible that the attacker temporarily blocks some inputs to an anonymity system, in order to ease the analysis of the other users. As we will see later, this technique is used to help attacks such as the (n-1)-attack.

Clogging Attack

A combination of a packet-counting and a denial of service attack leads to the clogging attack [70]. In this attack, the attacker wants to know the originator of a traffic flow. For this, he observes the message flow between two nodes AS_n and R (see Figure 4.1, dashed line from AS_n to A) and floods randomly-chosen nodes of the network with a large amount of traffic (see Figure 4.1, line from A to AS_1). If the traffic to node R collapses, it

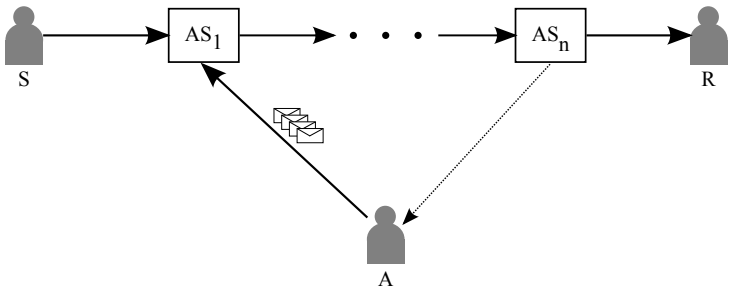

Figure 4.1: Setup for clogging attack

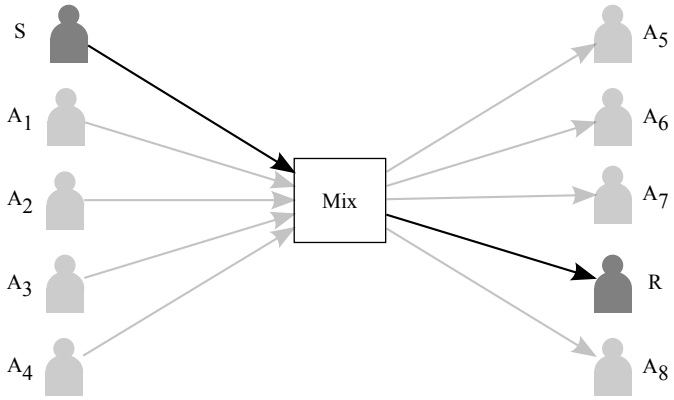

Figure 4.2: Setup for (n-1)-Attack

is highly probable that the flooded node is part of the communication path to node R. By successively choosing neighbors of the detected nodes, the originator S of the traffic can be revealed. This attack can only be successful if node S sends a high volume of traffic to the same node over an extended period of time, because traffic of other nodes could also be the reason for a collapse in traffic at node R.

(n-1)-Attack

The (n-1)-Attack targets Mixes and Mix networks [71, 72]. It is also known by the name of *blending attacks*. The attacker floods a Mix node with fake messages in order to trace a single message from an honest user. The setup of the (n-1)-Attack is shown in Figure 4.2. In this figure, the light gray inputs and outputs of the Mix belong to the attacker, while the dark gray input and output belong to the honest user. The attacker is able to recognize his

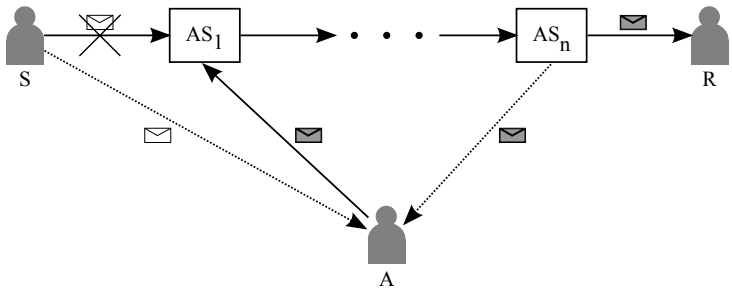

Figure 4.3: Setup for watermarking attack

messages when entering and leaving the Mix and therefore link the message from the honest user to its receiver.

To launch this attack, the attacker must be able to first delay or hinder other messages from being sent to the Mix and he must also be able to insert fake messages into the Mix. Some Mixes are able to detect (n-1)-Attacks and can react either by stopping the operation or by generating a response designed to confuse the attacker. For instance Red-Green-Black Mixes (RGB-Mix) [73] have a mechanism to detect such attacks. A RGB-Mix injects heartbeat messages addressed to itself by way of a path traversing some other Mix nodes. If an attacker tries to perform a (n-1)-Attack, the stream of heartbeat messages collapses. As reaction the RGB-Mix injects dummy messages at the output link of the Mix in order to confuse the attacker.

Watermarking Attack

Watermarking is also used to make traffic analysis easier. In this case, the attacker modifies a traffic flow to make it contain a specific pattern. If this pattern is detected at another position in the network, the two traffic flows are considered to be linked. For this attack, the attacker either belongs to the communication path, for example in a Mix network by controlling one Mix; or the attacker is able to block traffic between two nodes and insert modified packets (see Figure 4.3).

Watermarking is used for traffic analysis in [74, 75, 76]. A defense against such an attack is proposed in [77], where a method is introduced to detect and remove watermarks from traffic flows.

A variant of the watermarking technique is used in the *send n' seek attack* [59], which improves packet-counting attacks by inserting messages in the communication. In this case the attacker wants to uncover the identity of a recipient, for example in a privacy-protecting email system. The attacker sends a number of messages to the anonymous recipient. After having sent the messages, he tries to find the same number of messages at a receiving node. If he finds such a node, he assumes that this is the anonymous recipient.

Wei Dais' Attack on Traffic Shaping

The original attack is published online at [78] and it is referenced by Back et al. in [70]. The attack targets traffic shaping. Traffic shaping is used by some anonymity protocols to hide the real traffic, for example by always sending a constant volume of traffic. If there is not enough real network traffic, the anonymity system generates dummy traffic between its stations.

In the attack introduced by Dai, the attacker builds a route over stations of the anonymity system back to himself and increases his traffic as far as possible. At some point the bandwidth limit is reached. The attacker is now able to determine the amount of real network traffic that is sent between the stations of the anonymity system by removing its own traffic from the bandwidth limit.

Attack on Reservation in DC-net

The weak point of the DC-net protocol (see Section 3.2) is the reservation of communication rounds. During the reservation phase of the protocol, the users of the DC-net agree an order for communication, in which each user should only know his own position for sending. An attacker can use this reservation phase to disrupt the protocol.

By changing some bits randomly in his output message, an attacker can prevent other users from sending messages. Chaum proposes a way to find such disrupters. The honest participants use traps in order to find attackers. First, the honest participant fixes the index i of the round where he wants to lay the trap, a random message M and a symmetric key k. He then publishes the random message M and the index i, both encrypted with the secret key k. Later, during the execution of the reservation protocol, he

reserves the communication slot i in which he sends his random message M. If this message is disrupted by an attacker, he publishes the secret key k, allowing all participants to identify the attacker that has been caught by the trap. Zero-knowledge proof techniques can also be used to find disrupters in DC-net. This technique was introduced by Golle and Juels [79] and later used by Franck [80].

Attack on Traps in DC-net

The traps which have been introduced in the previous section introduce new vulnerabilities to the DC-net protocol. In an attack described in [55], the attacker builds a trap without reserving the corresponding communication round. If an honest participant P reserves this communication round by chance, the attacker can later open his trap. As the output sent by the honest participant is different from the random message in the trap, all participants have to reveal their secret keys. As result, the anonymity of P is broken.

4.2 Slotted Packet-Counting

The *slotted packet-counting attack*, as introduced in this section, was published by us in [12]. Common packet-counting attacks make strong assumptions about the setup, which can easily lead to incorrect conclusions, as explained in Section 4.2.1. To overcome these limitations, we propose the slotted packet-counting attack, which accounts for the variation of traffic load over time. We use correlation to express the relation between sender and receiver nodes. Our attack is applicable to many anonymity and unlinkability protocols, such as Tor and Mix networks (see Section 4.3). It assumes a passive attacker and works with incomplete knowledge of network traffic.

4.2.1 Motivation

Common packet-counting attacks, as introduced in recent literature (see [70, 59, 34]), are not efficient when a passive attacker, as described in Section 2.2, Definition 8, is assumed. In the following we briefly explain this type of attack, and show that in most scenarios packet counts at sender and receiver

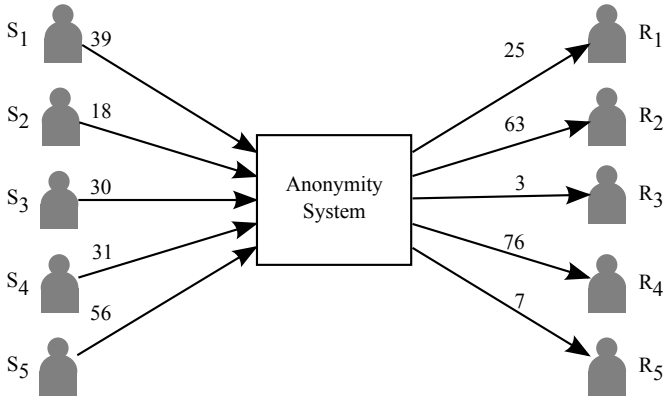

Figure 4.4: Example for packet-counting

side do not offer enough information to link a sender to the receiver of a communication.

The idea of packet-counting attacks, as described in the literature, is that the attacker can observe the sending and receiving of packets. The attacker counts these sending and receiving events and tries to find similarities in order to derive plausible pairs of communication partners. This is only possible if a relation between the similarity of the number of sending and receiving events and the probability for the associated communication link exists.

Assume that the attacker has no further information except the number of packets sent by some sender nodes $S_1, \cdots, S_n$ and received by receiver nodes $R_1, \cdots, R_m$. Then, the probability that a node S_i sent a packet to node R_j does not actually depend on how similar the number of sent packets by S_i and received packets at R_j are. The only thing one can say is that the probability becomes higher the more packets S_i sends and the more packets R_j receives provided that, at the same time, the number of sender and receiver nodes stays constant. This becomes clear when considering the following example.

Figure 4.4 shows an anonymity system with sender nodes $S_1, \cdots, S_5$ and receiver nodes $R_1, \cdots, R_5$, as it is seen by an attacker. For evaluation of attacks we show the same setting in Table 4.1, where additionally the communication partners and the corresponding number of packets are given.

Now, assuming that we have no further information except the number of packets sent and received, as in Figure 4.4, we can compute the probabilities for every node S_i communicating with R_j as relative frequencies. We write

Table 4.1: Communication partners

	S_1	S_2	S_3	S_4	S_5
R_1	0	0	25	0	0
R_2	32	0	0	31	0
R_3	0	3	0	0	0
R_4	5	15	0	0	56
R_5	2	0	5	0	0

Table 4.2: Likelihood for communication

	S_1	S_2	S_3	S_4	S_5
R_1	0.999	0.948	**0.994**	0.995	1.000
R_2	**1.000**	1.000	1.000	**1.000**	1.000
R_3	0.535	**0.281**	0.435	0.447	0.691
R_4	**1.000**	**1.000**	1.000	1.000	**1.000**
R_5	**0.837**	0.541	**0.741**	0.753	0.938

$c(S_i)$ to denote the number of packets that node S_i sends, and $c(R_j)$ for the number of packets received by node R_j respectively. We denote the number of all receiver nodes by n. The nodes $R_k, 1 \leq k \leq n$ together receive $\sum_{k=1}^{n} c(R_k)$ packets, and $R_k, 1 \leq k \leq n, k \neq j$ together receive $\sum_{\substack{k=1 \\ k \neq j}}^{n} c(R_k)$ packets. With this information we can calculate the probability of S_i having sent a packet to R_j by:

$$P(S_i \textit{ sent to } R_j) = 1 - \frac{\binom{\sum_{\substack{k=1 \\ k \neq j}}^{n} c(R_k)}{c(S_i)}}{\binom{\sum_{k=1}^{n} c(R_k)}{c(S_i)}} \,. \tag{4.1}$$

The resulting probabilities when using the values of Figure 4.4 are shown in Table 4.2. The bold entries in the table indicate where actually a communication took place (see also Table 4.1). The table shows that the more packets sent by a node S_i and the more packets received by a node R_j the higher is the probability that at least one packet was sent by S_i to R_j. Hence, correlation of sender nodes and receiver nodes only by counting packets is not very effective.

This negative effect is also explored in existing literature. Raymond [59] says that only if one node sends an unusual number of packets, can the attacker find the receiver of this unusual number of packets. Serjantov and

Sewell [34] stress that packet-counting only works if the whole connection is *lone*. This means that several connections using an anonymity system at the same time neither share incoming links nor outgoing links of the system, i.e. the number of packets counted for an incoming or outgoing link can be related to one unique connection. Aside from the fact that this is a strong assumption, it is not always possible for the attacker to find out whether a connection is lone or not. Therefore, using only the packet counts at sender and receiver side does not provide enough information to make a reliable mapping of sender and receiver nodes. In the next section, we show how to improve the attack by taking account for the alteration of packet counts on sender and receiver side over time.

4.2.2 Attack Details

Our slotted packet-counting attack builds on the idea of earlier packet-counting attacks. The main difference is that, with slotted packet-counting, we measure the alteration of traffic on sender and receiver sides over time. If the alterations on both sides correlate, the probability that they communicated increases.

To measure the alteration of the packet counts over time, the attacker counts the sending and receiving events in time slots $t_1, \cdots, t_m$. For every sender node S_i we build a random variable C_{S_i} with values $c_k(S_i)$,$1 \leq k \leq m$, giving the number of packets sent in time slot t_k. In the same way, we have a random variable C_{R_j} with values $c_k(R_j)$, $1 \leq k \leq m$, representing the number of packets received in time slot t_k at node R_j.

To link the packets in time slot t_k at the sender side to packets at receiver side, the attacker has to be aware of timing. Because packets need some time to transit through the anonymity system, he must shift the time slot t_k at receiver side by this delay. We give more details on this issue later in this section.

We can now calculate the correlation between the random variables of the sender nodes and the random variables of the receiver nodes in order to map them together (for details on probability theory see e.g. [81]). The correlation coefficient of two random variables gives the strength and direction of their linear relationship. Its value is in the interval $[-1, 1]$, where 0 stands for non-correlating variables, 1 and -1 for correlating variables. We are only interested in the cases where the correlation coefficient is near to

1, what means that the random variables have a positive linear relationship. These are the cases where it is highly probable that the corresponding nodes communicated with each other. The correlation coefficient of two random variables X and Y is calculated as follows:

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_x \sigma_y} = \frac{E((X-\mu_X)(Y-\mu_Y))}{\sigma_x \sigma_y} \ . \tag{4.2}$$

In order to calculate the correlation coefficient of two random variables C_{S_i} and C_{R_j}, we need the related expectation values μ_{S_i} and μ_{R_j} as well as the standard deviations σ_{S_i} and σ_{R_j}. We assume that the random variables C_{S_i} and C_{R_j} have a Gaussian distribution and justify this with the central limit theorem (see [81]). From this, we can calculate the expectation value μ_{S_i} as follows:

$$\mu_{S_i} = \frac{1}{m} \sum_{k=1}^{m} c_k(S_i) \ . \tag{4.3}$$

Additionally, we can compute the standard deviation σ_{S_i} by:

$$\sigma_{S_i} = \sqrt{\frac{1}{m} \sum_{k=1}^{m} \left(\mu_{S_i} - c_k(S_i)\right)^2} \ . \tag{4.4}$$

Having calculated the expected values and standard deviations for all sender and receiver nodes we can calculate the correlation between every pair of sender S_i and receiver R_j. In the following, we illustrate the calculations by means of an example.

Example

We use the network and traffic of Figure 4.4 but split the observations of the traffic into five time slots $t_1, \cdots, t_5$. Table 4.3 shows the slotted packet count for sent packets with $S_1, \cdots, S_5$; Table 4.4 for received packets with $R_1, \cdots, R_5$.

With these values we can calculate expectation values and standard deviations for all combinations of sender and receiver nodes using the formulae (4.3) and (4.4). Table 4.5 shows the correlation coefficients of sender and receiver nodes calculated with formula (4.2). From these values one can conclude that the nodes S_2 and R_1, S_1 and R_3, S_2 and R_4, S_4 and R_5

Table 4.3: Received packet count by time slot

	t_1	t_2	t_3	t_4	t_5
S_1	7	3	14	6	9
S_2	3	3	7	3	2
S_3	7	11	3	7	2
S_4	3	8	6	2	12
S_5	8	12	4	20	12

Table 4.4: Packet count for receiving per time slot

	t_1	t_2	t_3	t_4	t_5
R_1	6	9	1	7	2
R_2	8	11	18	5	21
R_3	3	0	0	0	0
R_4	8	15	13	26	14
R_5	3	2	2	0	0

communicated with high probability (bold entries in Table 4.5); this also agrees with the assumed communication pattern given in Table 4.1. The comparison with the results of common packet-counting in Table 4.2 clearly shows that much more precise results are obtained by taking correlation into account.

Parameters of the Attack

We have already shown the basic concept of slotted packet-counting attacks. This section covers some techniques for the application of the attack in practice.

The slotted packet-counting attack provides a means of determining whether a node S communicated with a node R. In the description of the attack we explained how the attacker applies the slotted packet-counting

Table 4.5: Correlation coefficients of sender and receiver nodes

	S_1	S_2	S_3	S_4	S_5
R_1	-0.938	-0.492	**0.961**	-0.385	0.547
R_2	**0.623**	0.252	-0.721	**0.843**	-0.536
R_3	-0.109	-0.172	0.155	-0.444	-0.302
R_4	-0.257	-0.147	0.147	-0.274	**0.846**
R_5	0.018	0.363	0.362	-0.296	-0.704

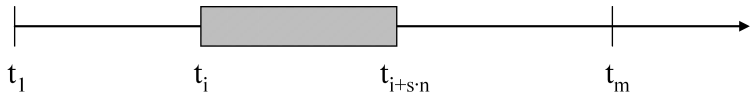

Figure 4.5: Locating communication slots

attack to a given set of communication slots. Here, we will give rules for choosing the communication slots that are used for the attack.

The problem is that the attacker has to find the points in time when S communicates with R. To do this, the attacker measures the packets sent at node S and received at node R during the time period, in which he guesses that a communication has occurred. He divides the observations in m time slots of length s and takes $n < m$ of these time slots for measuring the correlation. Therefore, he applies the slotted packet-counting attack for all intervals with the time slots t_i to $t_{i+n}, 1 \leq i \leq m - n$ (see Figure 4.5). A correlation coefficient close to one for any of the intervals indicates with high probability that there was a communication. If there exists more than one such interval, the probability that S sent packets to R becomes higher.

Slot size and number of slots The attacker has first to specify the values for size of time slots s and the number of time slots n. The optimal values for s and n mainly depend on the communication characteristics. Using longer time slots or a higher number of time slots does not necessarily mean that it is easier for an attacker to find a correlation. In the optimal case, the duration $s \cdot n$ of the observation is exactly as long as the communication period of the two target nodes. If the observation time becomes longer, the chance increases that the sender node sends packets to different receiver nodes or that another sender node interferes at the receiver side (giving more false negatives). On the other hand, a short observation interval makes it more likely that correlation between sender and receiver nodes occur even if there is no communication (more false positives). We show how to optimize the slot size and the number of communication slots used for a concrete attack scenario in Section 4.2.3.

Delay In order to map packets to the same time slots on sender and receiver side, the attacker has to be aware of timing. Packets traveling from the sender node S to the receiver node R need some time d to transit the network and

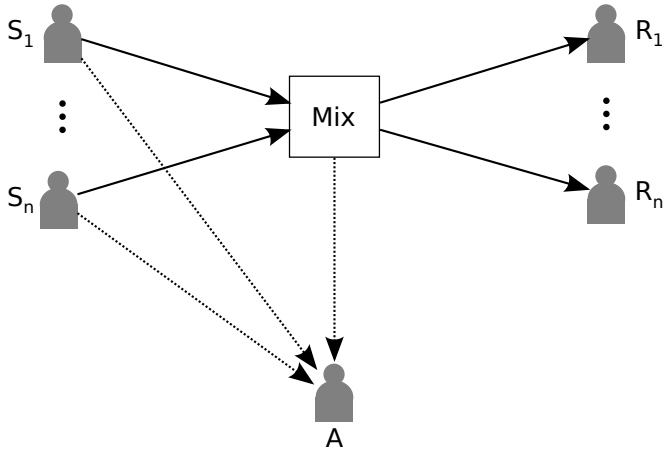

Figure 4.6: Setup for attacking Mixes

the anonymity system. In order to have a packet p that is sent at S and received by R in the same time slot t_k at S and R, the attacker has to shift the time slot on receiver side by the delay d. This delay d depends on the network, the network load, the positions of S and R in the network, and the anonymity system. To approximate d, the attacker can send test messages, allowing him to measure the time delay between sending and receiving a message while using an anonymity system. More information on timing of packets can be found in [82]. We will assume this timing in the application of the attack in Section 4.2.3.

4.2.3 The Attack in Practice

In this section, we apply the attack to real network traffic. As we have already seen, the efficiency of the slotted packet-counting attack depends on several parameters. We define these parameters and optimize them for the given traffic traces. It is important to note that different scenarios lead to different optimal parameters. We give a mechanism for finding these parameters that is applicable to all network traffic traces.

Setting

In order to define the parameters for the attack, we apply it to captured network traffic traces, namely the LBL-PKT-5 traffic traces [83] of TCP traffic. The traffic was captured at the Lawrence Berkely Laboratory from

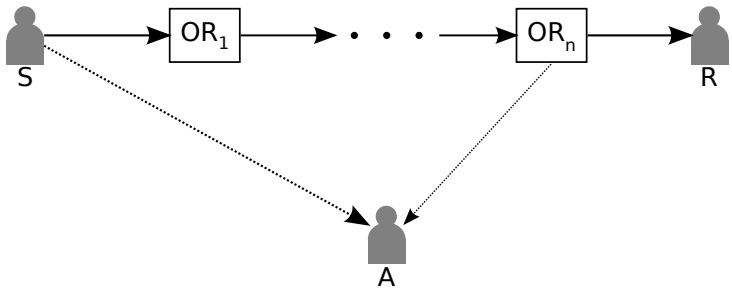

Figure 4.7: Setup for attacking Tor

14:00 to 15:00 on January 28, 1994. We assume two scenarios where we want to measure the correlation coefficient: in the first, we assume a Mix with different numbers of input and output links, where one slot refers to one communication round. The attacker receives all packets going to and coming from the Mix (see Figure 4.6). In the second scenario, we assume a low-latency anonymity protocol, such as Tor, where we use time intervals to define communication slots. In this case, the attacker only needs to capture the incoming traffic of the first onion router and the outgoing traffic of the last onion router on the path (see Figure 4.7).

We execute the attack with varying parameters:

Threshold The threshold for the correlation coefficient. If this threshold is exceeded, the algorithm assumes a communication link. We vary this parameter in the interval from 0.90 to 0.99.

Number of slots We vary the number of slots which are used for the measurement of the correlation coefficient from 3 to 500.

Size of slots For the Mix scenario we assume Mixes with 5 to 455 inputs and for the Tor scenario we use slots of size from 0.5 sec to 5 sec.

For evaluation, we start the algorithm with different input traces and every possible combination of parameters and use the mean of the different runs. In the following, we discuss the results of the measurements.

Identifying Parameters

We use the *F-Measure* [84] to identify optimal values for the size and the number of slots used for the measurement. The value of the F-Measure is in the interval $(0, 1]$, where higher values signify a higher accuracy. We use

the F-Measure to provide a trade-off between false negatives and the false positives that occur when only a few slots are used for the measurement. The F-Measure is calculated as follows:

$$F = \frac{2 \cdot tp}{2 \cdot tp + fn + fp},$$

where tp means true positive, fn means false negative, and fp means false positive.

The reason for false negatives is mainly interference from other communications, i.e. changes in communication links or nodes communicating with more than one node. This interference can be minimized by using a shorter period of time for the measurement.

Figure 4.8 and 4.9 show that the value for the F-Measure depends on the size and the number of communication slots. In both examples, we fixed the threshold for the correlation coefficient to 0.95. In the Mix scenario we vary the number of slots used and the size, i.e. number of incoming and outgoing links. The size of a Mix is defined in its specification and cannot be manipulated by the attacker. The F-Measure value reaches its maximum for 90 slots for a Mix of size 5 and around 10 slots for a Mix of size between 50 and 500. We achieved the best results for the Tor scenario at a slot size of 0.5 seconds and 13 slots. The F-Measure value becomes smaller when we increase the size of the slots, while the optimum number of measured slots stays in the region of 13 slots. We did not apply the attack with smaller slot sizes to the Tor scenario because we idealized the network scenario for the tests by removing network delays. In real network settings, delays and jitter can result in packet counts that are not precise enough if the slot sizes are too small.

Figure 4.10 gives an example in which an increase in the number of communication slots used for measuring the correlation coefficient does not lead to a better correlation. In the example, a fixed node S_i sends packets to a fixed node R_j. At some point, node S_i changes its communication partner to node R_k. This results in a decreasing correlation coefficient between the sending and receiving events of S_i and R_j.

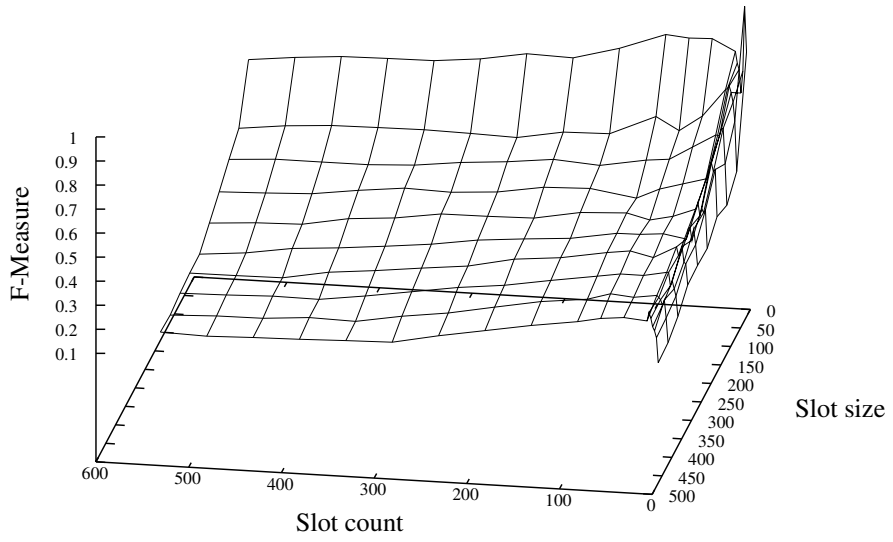

Figure 4.8: F-Measure of Mix scenario

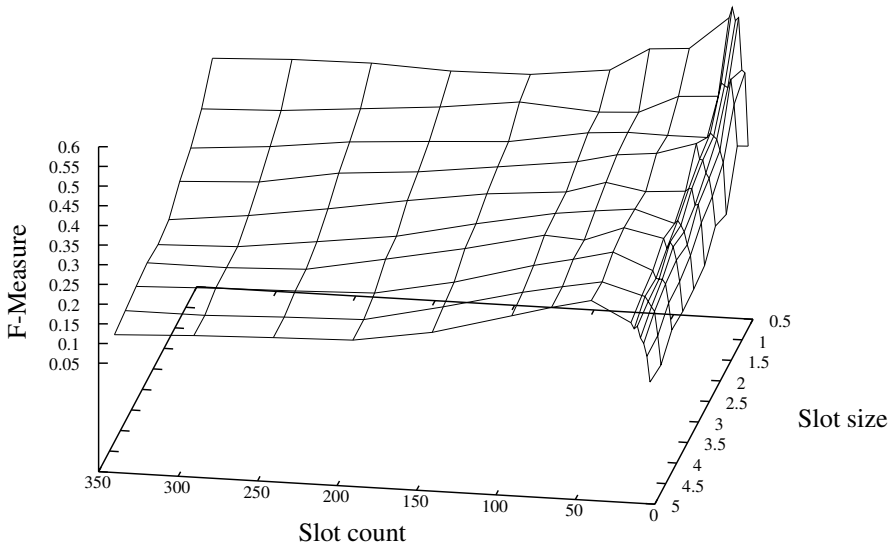

Figure 4.9: F-Measure of Tor scenario

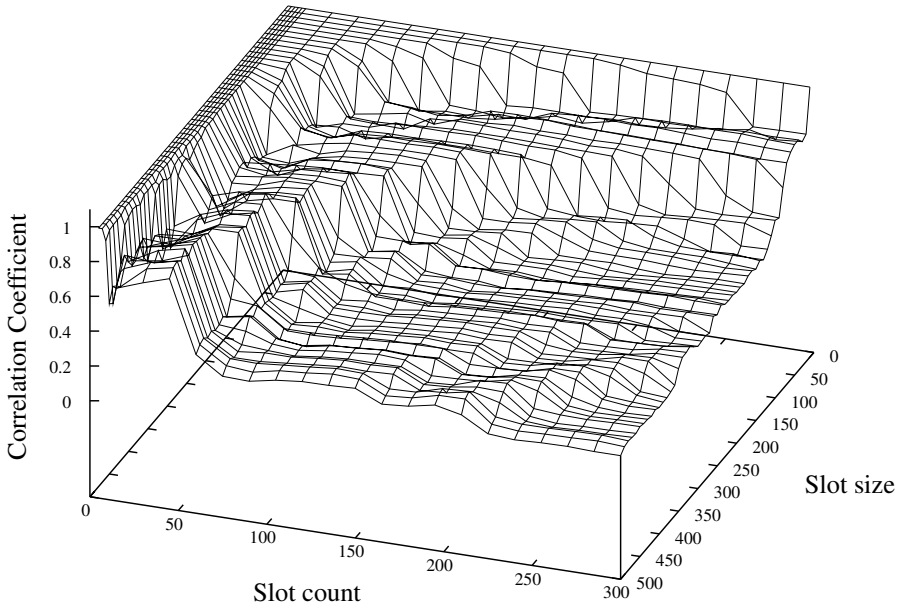

Figure 4.10: Correlation coefficient of nodes S_i and R_j

Discussion

The application of the slotted packet-counting attack to real network traffic traces shows the practicability of the attack. We showed how to adjust the parameters of the attack in order to improve the output of the attack. Further improvements in the parameters are possible: it might be better to use different parameters for different communication settings. For example, visiting a web site on the Internet produces a different number of packets from a VoIP phone call. For communication settings where a lot of traffic is produced, the attack will be more precise when using more slots. A large number of slots offers no useful results for settings where only few packets are sent over a short period of time.

As we have seen in the previous section the attack can fail in two ways: In the first case, there is a strong correlation between a sender node S_i and a receiver node R_j although there has been no communication between these nodes (false positive). One reason can be that the measurement uses insufficient information, i.e. the period of time covered by the measurement is too short. Another reason might be that the sets of sender and receiver nodes are too big, giving a higher chance of correlations between sender and receiver nodes that are not communicating. Hence, one aspect making this

attack effective is the minimization of the set of potential sender and receiver nodes. For instance by additionally using the timing of sending and receiving events, the set of potential sender and receiver nodes can be reduced. By combining these techniques we might gain better results for slotted packet-counting attacks.

The second manner of failure is when there is no correlation between nodes S_i and R_j even though these nodes are communicating (false negative). As seen in the previous section, this happens if there is interference from other nodes, i.e. there is a node S_k also sending packets to R_j, so that the number of received packets at R_j is related not only to the sending of S_i. In this case, the attacker can calculate the correlation between the sum of packets counted at the sender nodes S_i and S_k and the packets counted at the receiver node R_j. For performance reasons, this grouping of sender and receiver nodes should be limited to small groups.

Note that the problems addressed above also exist in the packet-counting attacks described in Section 4.2.1. However, slotted packet-counting has the advantage that false positives are less probable, because they only occur when the packet count is similar in every slot. Also, good results can be obtained by just observing the nodes of interest because the correlation coefficient needs only knowledge of the sending and receiving of these nodes. Only for more complex attacks, as mentioned above, is more information needed.

In general, this attack is applicable to all anonymity protocols where the attacker is able to detect sending and the receiving events. In more specific settings where there is only a limited group of users of the anonymity protocol (e.g. anonymity protocols for ad hoc networks such as [57], [85] and [86]), the slotted packet-counting attack is even more effective. The only way to prevent any kind of packet-counting for this class of protocols is to use *constant link padding*, which is not practicable in most settings because of the overhead it produces (see also [59]). What is more, as stated in Section 4.1.2, there exist methods to filter the padded traffic out of the stream. Other attacks, such as watermarking, might improve the slotted packet-counting attacks.

A similar approach to slotted packet-counting is proposed by [82]. The authors use cross correlation to map input and output traffic of Mixes, while we concentrate on the mapping of sender and receiver devices. They show the effect of network parameters on the attack but perform no optimization

of the attack parameters. We also give rules on how to find parameters for the attack in different network settings.

4.3 Efficiency of Attacks

In this section we show the influence on unlinkability of the attacks presented in the previous two sections on protocols for unlinkable communication. Table 4.6 gives the minimal degree of unlinkability that the system is able to guarantee when the attacker performs one of the given attacks (see Section 2.1 for the measures). In order to calculate the degree of unlinkability, we assume a network with m participants, where the Mixes have n in- and outputs, the paths in Tor consist of arbitrarily many onion routers and, in the case of a DC-net, the groups consist of n nodes. Most of the attacks need to observe the sending and receiving of several messages. We assume a strong attacker who is able to perform the attack in the best possible way, i.e. he is able to observe every message that is sent over an extended period. This gives the worst case degree of unlinkability. In practice, most of the attacks are not as effective as stated in the table, due to a shorter observation time and weaker attackers.

Some of the attacks are not able to compromise the unlinkability on their own but can be used to facilitate other kinds of attacks, such as the denial of service attack and delaying attack, which can help the (n-1)-Attack.

The table shows that Mixes and Tor are vulnerable to most kinds of traffic analysis attacks, where DC-net offers at least a degree of unlinkability of $\log_2 n$. The unlinkability of Mixes and Tor basically relies on the fact that traffic analysis attacks are hard to perform and, in most cases, the attacker is not able to collect enough information to make a reliable mapping of sender and receiver nodes.

The DC-net approach is not vulnerable to most of the passive attacks. Only communication pattern attacks have a chance of success, assuming that the attacker has previous knowledge on the participants of the DC-net group. For example, this is the case when the attacker is able to detect communication patterns for which only a subset of the DC-net group members might be the originator or the receiver, e.g. the attacker detects a protocol of a service which only some of the group members offer. There also exist attacks on round reservation and superposed sending, which can either be

Table 4.6: Influence of attacks on protocols for unlinkable communication with n users in networks with m nodes

	Mix	Tor	DC-net
Packet-counting attack	0	0	$\log_2 n$
Slotted packet-counting attack	0	0	$\log_2 n$
Message coding attack	$\log_2 m$	$\log_2 m$	$\log_2 m$
Packet volume attack	$\log_2 m$	$\log_2 m$	$\log_2 m$
Timing attack	0	0	$\log_2 n$
Communication pattern attack	0	0	0
Intersection attack	0	0	$\log_2 n$
Denial of service attack	no service	no service	no service
Replay attack	$\log_2 m$	$\log_2 m$	$\log_2 m$
Delaying attack	$\log_2 m$	$\log_2 m$	$\log_2 m$
Clogging attack	0	0	$\log_2 n$
(n-1)-Attack	0	n.a.	n.a.
Watermarking attack	n.a.	n.a.	n.a.
Wei Dais' attack	n.a.	0	n.a.
Attack on reservation	n.a.	n.a.	no service
Attack on traps	n.a.	n.a.	0

used to disrupt the execution of the protocol, or to reveal a communication link.

Additionally, the weaknesses in the protocol design also the implementation of the protocols might introduce new vulnerabilities. For example, the attack of Bauer et al. [49] exploits the mechanism of the path-building algorithm in Tor. In order to obtain communication paths with good performance, the Tor implementation prefers onion routers with a good performance. Therefore, an attacker only needs to insert few onion routers with high performance in order to have a good chance that both the first and the last onion router on the communication path are under his control. The fact that Tor builds a new communication path every minute additionally increases the chance that the attacker is able to reveal a communication link.

4.4 Conclusion

This section showed that all protocols, where both endpoints of a communication are observable by an attacker, are vulnerable to traffic analysis attacks. These attacks are based mainly on statistical analysis of captured traffic information, such as the number of packets sent or received by a

node. By combining different observations over an extended period of time, the attacker is able to reveal communication links. These passive attacks can be made more effective by combining them with active attacks, i.e. attacks where the attacker actively manipulates traffic and parts of the network.

The DC-net protocol is not affected by most of these traffic analysis attacks, because the attacker is not able to detect the endpoints of the communication; only communication pattern attacks have a chance of success. However, DC-nets have some specific weaknesses, such as the reservation of communication slots and the superposed sending. Because of the anonymous sending, it is hard to detect disrupting attackers. The techniques presented for the detection of disrupters introduce a chance for the attacker to reveal the sender of a message.

To summarize, the weakness of the presented proxy-based protocols is that the attacker gains information about the communication links by observing the communication. In DC-net, he might only be able to detect communication patterns, but there also exist specific attacks on DC-nets. The other problem of DC-nets is the poor scalability, which makes them unsuitable for most communication scenarios.

The following chapter introduces a new protocol for unlinkable communication, where passive observations provide no information about the communication link.

Chapter 5

Unlinkable Communication

This chapter builds on the work described in [13, 14, 11, 15]. There exist several protocols that claim to offer sender or receiver anonymity, which implies unlinkability of sender and receiver. We presented these protocols in Chapter 3. As we have seen in the previous Chapter 4, these protocols have serious weaknesses. The proxy-based approaches suffer from the fact that the end points of a communication are visible to a strong passive attacker. Therefore, it is possible to apply traffic analysis attacks to this type of protocol. In the dining cryptographers protocol, these attacks are not applicable. However, besides the fact that this protocol is very inefficient, it also introduces new flaws due to the need for round reservation. The attacker is either able to attack the round reservation directly or he can attack the countermeasures to these attacks.

To overcome the problems in the presented protocols, this section introduces an improved protocol for unlinkable communication. In this protocol, the attacker is able to detect the nodes that are sending messages, but not the nodes that are receiving messages. Consequently, he is not able to apply traffic analysis attacks to this protocol. By using efficient cryptographic techniques, we can minimize the delay due to the use of the protocol. The message overhead is minimal with respect to the minimum guaranteed degree of receiver anonymity attained. We will prove that the protocol presented in this chapter offers unbreakable unlinkability under the presence of a strong passive attacker who is able to observe any transmission in the network. This is because the attacker is not able to infer additional information concerning the communication link by observation.

The rest of the chapter is organized as follows: in the first part of this chapter we introduce the protocol for unlinkable communication. The second part consists of a security analysis of the protocol. We prove that the attacker is not able to break the unlinkability by passive observation of network traffic. Finally, we show how to apply the protocol in practice.

5.1 Protocol

In this section we introduce a protocol for unlinkable communication for which we can prove that a strong passive attacker is not able to break the unlinkability and receiver anonymity. The major components of the protocol are published in [14, 11, 15]. We assume that the network topology is known to the user, i.e. every user has enough information to build a route to any destination node. In Internet scenarios this can easily be managed by a dedicated directory service (e.g. a central server), but P2P techniques are also applicable. We also assume that all nodes own a pair of public and private keys and that the public keys are accessible to other users of the network. This work does not discuss how this may be realized. These asymmetric keys will be used for secret key-exchange. We do not use the Diffie-Hellman key-exchange [87] for key-exchange because the protocol is susceptible to man-in-the-middle attacks. To keep the computational overhead small, we use efficient symmetric encryption during the communication phase.

Our protocol is based primarily on layered encryption, which is used while sending messages over a path made up of several routers. The destination of the communication is not at the end to the path, as it is in Tor for example, but at a random point on the communication path. In Figure 5.1 the receiver node N_t is placed at the fourth position on the path. The other nodes on the path are also users of the protocol. This means that an attacker is not able to map source and destination of a communication by traffic analysis attacks. Indeed, we can prove that an attacker is not able to obtain any information about the communication link while observing the network communication, given the assumption that the nodes on the path are chosen properly. In this chapter we denote the sending node by N_1, the receiving node by N_t and the last node on the communication path by N_{t+e}, where the indices represent the position on the path.

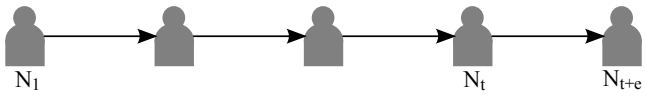

Figure 5.1: Communication path

In Section 5.1.1, we describe the protocol for exchanging the symmetric keys on the communication path. Following this, we detail the protocol for communication. We also introduce functionality for reverse communication, i.e. where the destination node sends reply messages to the source node. Finally, we give rules for the selection of communication paths. This is important because bad selections can leak information about the communication links.

5.1.1 Key-exchange

Because our protocol for unlinkable communication uses layered encryption, we have to distribute keys among the nodes on the communication path. We have assumed that every node on the communication path owns a pair of public and private keys, where the public keys are accessible by other users of the protocol. This work does not discuss the key distribution method. Details of this topic can be found for example in [88].

The key-exchange process is initiated by the node N_1 when it wants to send a message to the destination node N_t. N_1 generates secret keys K_i and seeds S_i for the generation of additional secret keys, $2 \leq i \leq t + e$, for all nodes on the path. As well as generating these keys, N_1 randomly chooses channel identifiers C_i, which are used to identify communication channels while sending data messages. N_1 needs to store these values for all nodes on the path in order to build communication messages.

Having generated these values, N_1 can build the key-exchange message. It first builds the message M_{t+e} by encrypting the concatenation of the identifier ID_{t+e} of the last node on the path N_{t+e} and the secret key K_{t+e} with the public key PK_{t+e} of N_{t+e}. N_1 also appends the channel identifier C_{t+e}, the seed for generating additional secret keys S_i and a flag F_{end}, that indicates that the end node has been reached, encrypted with the secret key K_{t+e} to this encryption. This results in the following message for M_{t+e}:

$$M_{t+e} := [ID_{t+e}, K_{t+e}]_{PK_{t+e}}[C_{t+e}, S_{t+e}, F_{end}]_{K_{t+e}}$$

The messages M_i for $2 \leq i \leq t + e - 1$ are recursively constructed by encrypting the identifier ID_i and the secret key K_i with the public key PK_i. The concatenation of the channel identifier C_i, the seed S_i, the identifier of the next node on the path ID_{i+1} and M_{i+1} is encrypted with the secret key K_i. Both ciphertexts are concatenated to produce the message M_i:

$$M_i := [ID_i, K_i]_{PK_i}[C_i, S_i, ID_{i+1}, M_{i+1}]_{K_i}$$

Node N_1 now sends the key-exchange message M_2 to N_2. N_2 decrypts the first part $[ID_2, K_2]_{PK_2}$ of the message M_2 with its private key SK_2. If N_2 is able to decrypt this part correctly, the identifier ID_2 matches its own identifier. It uses the secret key K_2 to decrypt the rest of the message, which contains the channel identifier C_2, the seed S_2, the node identifier of the next node ID_3 and the message for that node M_3. The identifier of the predecessor node ID_{i-1}, successor node ID_{i+1}, the secret key K_2, the seed S_2 and the channel identifier C_2 are stored by the node N_2. In addition to the channel identifier, N_2 stores a counter MC_2 and initializes it to 0. During the communication, N_2 also stores a secret key that is derived from a pseudo-random number generator. The generated secret key is used only to decrypt a single message. N_1 and N_2 synchronize these one-time secret keys by using the message counter MC_2. The one-time secret key related to the message counter MC_i is denoted by K_{MC_i}. This message counter and the one-time secret key are used to prevent attackers from replaying messages on the path. The process of decrypting, processing and forwarding messages is repeated until the message M_{t+e} is processed by the last node on the path N_{t+e}. The flag F_{end} indicates that the message reached the end of the path and does not need to be forwarded. By this stage N_1 has shared secret keys, seeds, channel identifiers and message counters with all nodes on the communication path.

We use fixed packet lengths for the key-exchange messages in order to prevent an attacker from receiving any information concerning the position of a particular node on the path, by eavesdropping on its messages. This will not help to increase the unlinkability in the presence of a strong passive attacker that is able to observe every communication in the network; however, it complicates the analysis for weaker attackers. To keep a constant message size for the key-exchange message, we add dummy bits at the end

of the message each time a layer of encryption is removed. If block ciphers, such as AES [89] or DES [90], or one-time pads [90] are used for symmetric encryption, these dummy bits will not affect the encryption of the rest of the message.

5.1.2 Communication

In order to send data messages from node N_1 to N_t, a communication path has to be established by means of a key-exchange message. Following such a key-exchange message, N_1 shares a secret key K_i, the seed S_i for generating additional secret keys, a channel identifier C_i and a message counter MC_i with every node N_i, $2 \leq i \leq t + e$, on the communication path.

For sending data D to the destination node N_t, node N_1 wraps D in a data message using layered encryption. We split the building of this data messages into two parts: first, we show how the path information is wrapped into the message; then, we show how the data part is layer-encrypted. The concatenation of these two parts results in the data message.

The building of the path information of the data message starts with the end node. First, we build the encryption of the message counter MC_{t+e} and the channel identifier C_{t+e} as the initial path information part P_{t+e} of the message:

$$P_{t+e} := [MC_{t+e}, C_{t+e}]_{K_{t+e}}$$

To this path information, we recursively add the identifier C_i and counter MC_i and encrypt it with the secret keys K_i for all $2 \leq i < t + e$:

$$P_i := [MC_i, C_i]_{K_i}[P_{i+1}]_{K_i}$$

When using a block cipher such as AES and DES, the message counter and the channel identifier should be in the same encryption block. The reason for this is to prevent an observing attacker from simply linking messages to one communication by comparing the path information parts of the messages. By inserting the incrementing message counter, the encryption of the path information changes when sending different messages over the same path.

The data part of the message is also layer-encrypted, but starting with the one-time secret key K_{MC_t} of the target node N_t. The initial message

also contains a flag F_{dest} that indicates that the message has reached the target node. It is built as follows:

$$D_t := [F_{dest}, D]_{K_{MC_t}}$$

For $2 \leq i < t$ the data message is built with:

$$D_i := [D_{i+1}]_{K_{MC_i}}$$

Given these two parts, the message M_i can be built by concatenating the path information P_i and the data part D_i:

$$M_i := P_i, D_i$$

N_1 builds the message M_2 and sends it to node N_2. Every node N_i for $2 \leq i \leq t + e$ that receives a message M_i first opens the initial part of P_i by decrypting $[MC_i, C_i]_{K_i}$ with its secret key K_i. N_i processes the rest of the message only if the channel identifier matches that stored locally at N_i and the message counter MC_i in the message is greater than the locally stored counter. This message counter is used to prevent an attacker from replaying messages. Additionally, N_i is able to synchronize the one-time secret key K_{MC_i} with N_1. It decrypts the remainder of P_i with its secret key, yielding P_{i+1} and the data part D_i, with the synchronized one-time secret key K_{MC_i}, which results in D_{i+1}. N_i uses these parts to build the new message M_{i+1} and forwards this message to the successor node.

When the message reaches the destination node N_t, the decryption of D_1 contains the flag F_{dest} which indicates that the destination node has been reached. The content of the message is now in plaintext. In this case, N_t builds D_{i+1} by taking random bits. All nodes N_i for $2 \leq i < t + e$ forward the new message M_{i+1} that contains P_{i+1} and D_{i+1} to the next node on the path N_{i+1}. The end node N_{t+e} discards the message.

In the same manner as for the key-exchange, we can arrange a fixed packet length during the communication. First, we have to limit the amount of data D that can be sent with one data message. We can fix the length of the message parts P_i and D_i by adding dummy bits, in order that they reach a constant length. If block ciphers or one-time pads are used, these dummy bits do not influence the rest of the encryption.

5.1.3 Response Channel

With the basic communication protocol only communication from node N_1 to N_t is possible. Here, we present a procedure for sending messages back from node N_t to N_1, while keeping the destination node N_t anonymous.

The response channel is based on a ticket system, where node N_1 generates tickets for response messages. A ticket in our case is basically a blank data message that is filled with the response by N_t. It is sent on the path over N_t to N_{t+e} and bounced back to N_1. All other nodes perform the same actions as before: they check the message's channel identifier and message counter; they decrypt the rest of the message and forward it to the next node on the path. An attacker is not able to distinguish between such a ticket and a normal data message, and because of the layered encryption, he is not able to detect when N_t replaces the blank message with its reply.

Again, we split the message into two parts, a path and a data part. The path part of the ticket message is recursively built starting with the path B which goes from N_{t+e} to N_1. In order to receive messages, N_1 needs to generate a secret key K_1, a seed S_1, a channel identifier C_1 and a message counter MC_1. With this information N_1 can build the message for reverse communication starting with B_1 and ending with B_{t+e-1} with:

$$\begin{aligned} B_1 &:= [MC_1, C_1]_{K_1} \\ B_i &:= [MC_i, C_i]_{K_i}[P_{i-1}]_{K_i} \end{aligned}$$

where $2 \leq i < t + e$. The whole path part is constructed in a similar manner to a common communication message starting with P_{t+e} and ending with P_2:

$$\begin{aligned} P_{t+e} &:= [MC_{t+e}, C_{t+e}]_{K_{t+e}}[B_{t+e-1}]_{K_{t+e}} \\ P_i &:= [MC_i, C_i]_{K_i}[P_{i+1}]_{K_i} \end{aligned}$$

where $2 \leq i < t + e$.

The data part is built in the same way as a data message. The only difference is the flag F_{ticket} that indicates that this is a ticket that can be used for sending a response. The data only consists of dummy bits D_{dummy}:

$$\begin{aligned} D_t &:= [F_{ticket}, D_{dummy}]_{K_{MC_t}} \\ D_i &:= [D_{i+1}]_{KMC_i} \end{aligned}$$

for $2 \leq i < t + e$.

N_1 sends the ticket message M_2, built by the concatenation of P_2 and D_2, to N_2. All nodes N_i process the message M_i in the same way as for data message. The only exception is N_t, which unwraps the last layer of encryption of the data part. The flag F_{ticket} indicates that N_t can put its response $D_{response}$ encrypted with the one-time secret key K_{MC_t} in the data field. The data part of the resulting message is:

$$D_{t+1} := [D_{response}]_{K_{MC_t}}$$

N_t forwards the message M_{t+1} along the path to the end node N_{t+e}. All nodes on the way to the end node perform the same actions as before: They decrypt both the data and the path part of the message and forward the new message to the successor node. By decrypting the path part of the message the end node knows that the message has to be bounced back to N_1. For this N_{t+e} uses the path field B_{t+e-1} to send back the message.

$$M_{t+e+1} := B_{t+e-1}, D_{t+e+1}$$

All nodes that receive the backwards message forward the message back along the path until $M_{2\cdot(t+e)-1}$ reaches N_1. To obtain the data $D_{response}$ that was sent by N_t, N_1 has to unwrap all layers of encryption starting using the one-time secret keys K_{MC_2} through $K_{MC_{t+e}}$ then back to K_{MC_t}.

There are basically two different ways to allow for multiple reverse messages being sent in response to a single forward message. We present two ways to implement this feature: in the first approach, node N_1 constantly sends ticket messages to N_t until N_t sends a message back that indicates the end of the transmission. The advantage of this approach is that there is practically no delay after the first reply message has been received. In the second approach, N_t includes a flag in the response message that indicates the need for a new ticket. An advantage of this approach is that the amount of replay messages sent is minimized.

Unlinkability and receiver anonymity during backwards communication are guaranteed by the use of layered encryption, which ensures that the appearance of the message changes at every communication hop. As a result, the attacker is not able to detect when the dummy message is replaces by the response. Because all nodes perform the same calculations, the timing of packets at every node is basically the same. The only difference is that node

N_t also has to prepare the response message. We can counter this problem by first sending an announcement to N_t so that it has time to prepare the response in advance.

5.1.4 Path Selection

As defined in Section 2.2, we assume that the attacker knows the system being used. For this reason, we assume that the attacker knows the algorithm for choosing the nodes on the path, which builds the unlinkability set and the receiver anonymity set. The example of Tor highlights the dangers that can arise when choosing nodes. The attack of Bauer et al. [49] exploits the mechanism of Tor which tries to choose high-performance Tor nodes for communication. By inserting only few Tor nodes with high performance under the control of the attacker, he has a good chance of inferring communication links.

In order to avoid these kind of attacks a good protocol should not prefer any nodes' property, such as high performance. In the case of Tor, the Tor nodes are only relays but cannot be users of the system. In contrast, in our protocol all these nodes can be the destination of a communication. For this reason, all nodes on the path, including the end node, should be equally likely to be the destination of the communication. If this is not the case, the attacker might be able to map the source to the destination by reproducing the communication path. We identify the following criteria which an attacker might use to group the nodes:

- Services offered by the nodes
- Performance of nodes
- Location of the nodes in the network

In the first case, different services might result in different traffic patterns. An attacker might map these traffic patterns back to a special service. If only a subset of the nodes on the path offers this service, the attacker can reduce the receiver anonymity set and unlinkability set to these nodes. In the second case, if only high-performance nodes are used to build communication paths, the attacker can identify the destination node if this is the only node with lower performance. And finally, in the case of ad hoc networks, the location of nodes can give additional clues concerning the identity of the receiver. To

prevent this information from reducing the degree of unlinkability, the path has to be chosen in such a way that the nodes on the path are similar with respect to services offered, performance and location.

If we assume that the nodes on the path are selected such that the probability of selection is equal for all nodes in the network, we can calculate the information that the attacker obtains by observation. We define n as the number of nodes in the network and $t + e$ as the length of the path over which the sender sends messages to the target node. With these parameters we can calculate how much information about the target node $N_t \in N$ a strong passive attacker can maximally receive if he knows the complete path $p_i \in P$ beginning at the sender node $N_1 \in N$ and ending at the last node N_{t+e}. Let $E(P) = \{e(p_1), \ldots, e(p_u)\}$ be the event set of P and $E(N) = \{e(N_1), \cdots, e(N_n)\}$ the event set of N.

$$I_{E(P),E(N)}(e(p_i); e(N_t)) = \log_2 \frac{P(e(N_t)|e(p_i))}{P(e(N_t))}$$
$$\log_2 \frac{\frac{1}{n}}{\frac{1}{t+e}} = \log_2 \frac{t+e}{n}.$$

If all nodes of the network belong to the communication path, that is, if $t + e = n$, the attacker receives no information on the target node.

If we use the definition of Section 2.1.3 and restrict the unlinkability set to the nodes in p_i, then the probability of any node N_i being the target node is $P(e(N_i)) = \frac{1}{p}$. As a result, the degree of unlinkability and the degree of receiver unlinkability is $\log_2 p$. For the calculations we have assumed that the probability of being the destination node is the same for all nodes in the network. If the attacker has previous knowledge that helps to adjust these probabilities, obviously the degree of unlinkability and receiver anonymity decreases. In any case, the attacker receives no additional information by observing the execution of the protocol other than by being able to identify the nodes on the communication path.

To avoid intersection attacks (see Section 4.1.1), i.e. attacks where the attacker intersects two different communications, there must be no paths that share same nodes. The set of nodes on a communication path has to be disjunct to the sets of all other communication paths. From this it follows that for sending several messages to the same destination node N_t the same communication path p_i has to be used. Additionally, for all other nodes N_j

that appear on this communication path p_i, the same communication path p_i has to be used if N_1 wants to send a message to one of the nodes N_i.

For sending messages from N_t back to N_1, only the response channel described in Section 5.1.3 must be used. If N_t breaks this rule by building a new path for sending messages to N_1 the attacker might be able to intersect both communications.

We can implement these constraints on the building of communication paths for Internet communication by choosing the nodes on the communication path randomly, and by using existing communication paths if the destination node already appears on one of them. For P2P communication the communication path can also be chosen randomly. However, we cannot easily implement this mechanism for ad hoc communication. Because the nodes might have no direct communication links to other nodes, the messages would have to be routed over additional intermediate nodes. This would cause longer communication paths, with not all nodes on the communication path contributing to the unlinkability set. What is more, longer communication paths give a higher chance of transmission errors. One way to implement an equal-probability choice of nodes in such a scenario is by first choosing the shortest path from source to destination and then expanding this path by appending nodes until it reaches a fixed length. The application of the protocol in different scenarios will be discussed in more detail in Section 5.3.

5.2 Analysis

In this section we give an analysis of the protocol for unlinkability and receiver anonymity that was described in Section 5.1. In the first part, we measure the performance of the protocol, which is composed of the induced message overhead and the computational overhead due to cryptographic functions.

An analysis of the unlinkability and receiver anonymity follows in the second part. We show that a strong passive attacker is able to detect which nodes belong to the communication path. Provided that the protocol is used properly, the attacker is not able determine which of these nodes is the destination node.

5.2.1 Performance

We separate the analysis of performance of the protocol into two parts: in the first part, we analyze the message overhead, i.e. how many additional messages are sent during the communication. The second part consists of an analysis of computational overhead by use of a prototype implementation of the protocol.

Message overhead

The protocol entails message overhead by exchanging symmetric keys through the use of a key-exchange message and during the communication because the messages are sent over the complete communication path, which generally extends beyond the intended destination node. In the following, we give details of the message overhead.

Key-exchange Key-exchange messages are sent only once for a communication path. Before the communication starts, the sending node N_1 builds a key-exchange message and sends this message hop by hop over all nodes on the communication path. Therefore, if the path consists of l nodes, an additional l messages must be sent.

The size of the key-exchange message depends on the algorithms used for the symmetric and asymmetric encryptions and also on the length of the communication path. In our prototype we use RSA with public key length of 162 bytes and private key length of 635 bytes for asymmetric encryption. If the messages are encrypted with the public key this results in a block length of 128 bytes. For a path of length l, the protocol generates a key-exchange message of length $l * 128$ bytes.

Communication phase For Internet communication, we are able to send the message directly to the destination node. Therefore, if instead we build a path over $l - 1$ other nodes, the message overhead during communication is $l - 1$. Together with the key-exchange, this results in a message overhead of $l + i \cdot (l - 1)$ messages when N_1 sends i messages to the destination node. Because l is constant and the unlinkability set consists of l members at a path length of l the message overhead increases linearly with the unlinkability set.

For other scenarios, where some of the nodes that are used to build the unlinkability set are necessary in order to connect to the destination node, such as in ad hoc communication, the message overhead decreases. But, in the worst case, the message overhead is still linear.

In all communication settings, the overall message overhead is in $O(l)$, where l is the size of the unlinkability set.

We can prove that this message overhead cannot be bettered if receiver anonymity is to be achieved.

Theorem 1. *Under the assumption that a strong passive attacker is able to observe every communication in the network, it is not possible to achieve a degree of receiver anonymity of* $\log_2 l$ *or better by sending fewer than* l *messages.*

Proof. We prove the theorem by contradiction. Assume that there is a protocol, that needs $n < l$ messages to achieve a degree of receiver anonymity of at least $\log_2 l$. From this it follows that fewer than l nodes in the network receive a message. Because the attacker is able to observe every communication in the network, he detects all n nodes that received a message. All other nodes in the network did not receive any message, so that the attacker is able to remove these nodes from the receiver anonymity set. As result the receiver anonymity set consists of only $n < l$ nodes. It follows that the degree of receiver anonymity is $d \leq \log_2 n < \log_2 l$, which contradicts the assumption. □

Computational overhead

We measure the computational overhead with the help of a prototype implementation of the protocol. The prototype was implemented in Java. We use AES with a key length of 16 bytes for the symmetric encryption, and RSA with a private key length of 635 bytes and public key length of 162 bytes. The reference hardware was a Dell Latitude D610 Laptop with a 2GHz Intel Pentium M processor, 1GByte main memory and Windows XP SP2. To evaluate the computational overhead, we measure the time required to generated and process key-exchange and communication messages. We average the results over 10^4 runs.

Firstly, we measure the performance of a key-exchange. Figure 5.2 shows how long it takes to generate and to process a key-exchange message. The

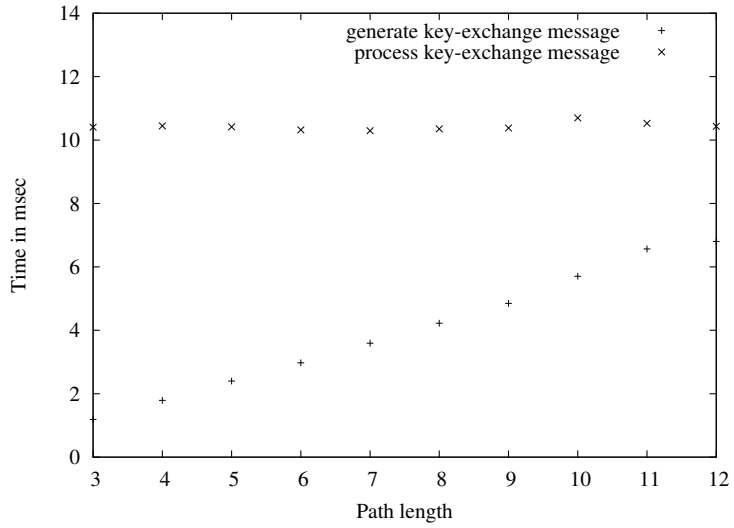

Figure 5.2: Performance of key-exchange

time required to generate a key-exchange message strongly depends on the length of the path, because the sending node has to perform one asymmetric encryption and one symmetric encryption for each node on the path. The growth of the time for generating key-exchange messages is nearly linear. The reason for this is that, if we add one node to the path, the sending node has to perform exactly one more asymmetric encryption and it also has to encrypt the body of the message symmetrically. In contrast to the generation of a key-exchange message, the processing does not increase noticeably when the communication path increases. All nodes only encrypt a small constant part of the message asymmetrically. As the communication paths become longer only the body of a key-exchange message grows linearly. For the body of the key-exchange message the nodes use efficient symmetric encryption with the exchanged keys while the part that has to be decrypted asymmetrically keeps constant. Therefore, the duration to process a key-exchange message does not grow significantly. In our measurements, the time to process a key-exchange is nearly constant. It increases linearly from 10.406 ms at a path length of 3 up to 10.432 ms at a path length of 12.

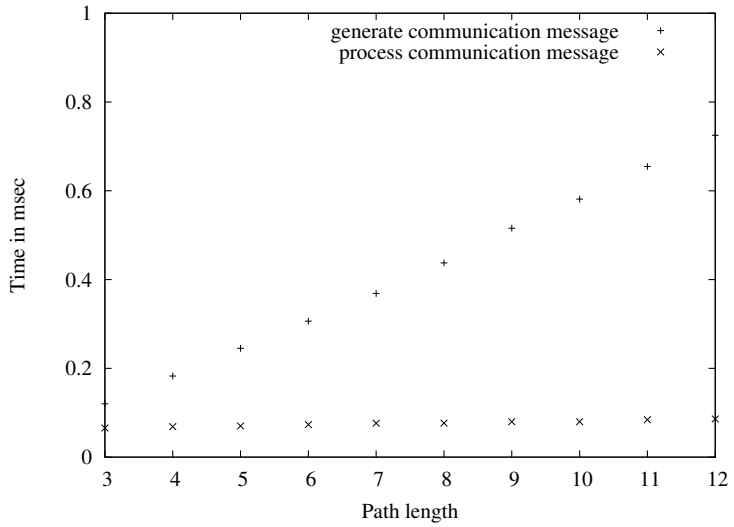

Figure 5.3: Performance of communication

In the same way, the time to generate a communication message increases when the communication path becomes longer. For the measurements in Figure 5.3, we assume a communication message with 1 KByte of data. The sending node encrypts the message using symmetric encryption once for every node on the path. The message itself only grows by 16 bytes if the communication path grows by one node. For that reason the time taken to generate a communication message is nearly linear in the length of the communication path. The processing of a communication message grows only slowly. Because the communication messages grow by 16 bytes for each node on the path, there is a minimal increase in the time that is needed for decryption. In our measurements the duration grows linearly from 0.066 ms for a path of 3 nodes up to 0.086 ms for a path of 12 nodes.

Figure 5.4 shows the throughput that the prototype implementation achieved during the measurements. To illustrate these values, we compare them with the throughput in non-anonymous Internet and wireless communication. In the case of Internet communication, the bottleneck is at the end-user side. Download rates in Germany are, at the time of writing this thesis, up to 32 MBit/sec and upload rates up to 1 MBit/sec (see e.g. [91]).

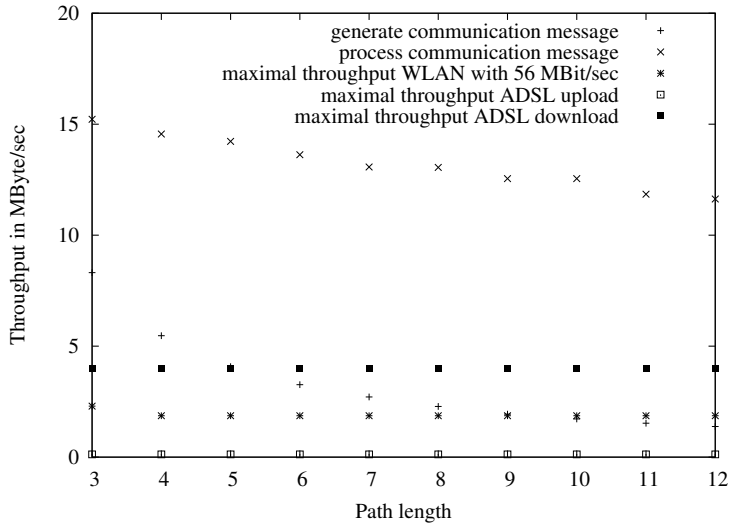

Figure 5.4: Throughput during communication

At a path length of 6, the throughput for generating communication messages falls below the theoretical best throughput of ADSL downloads, while the processing of messages always has better throughput. On the other hand, the throughput while uploading data with ADSL is always smaller than that for generating and processing messages.

For the ad hoc scenario, we compare the throughput of the protocol with the throughput of a WLAN. The common IEEE 802.11g WLAN standard [92], with a link rate of 54 MBit/sec, offers a maximal real throughput of 5.6 MByte/sec for one-hop communication. For multi-hop communication, the maximum throughput falls to one half for two-hop communication and one third for more than two-hops [93] (as in Figure 5.4). Even at a path length of 12, the protocol offers a better throughput for processing communication messages than can be transmitted over a WLAN. In our measurements the throughput of generating communication messages falls below the theoretical maximum throughput of WLAN at a minimum path length of 10.

In summary, the performance of the protocol is sufficient for use in the scenarios presented.

5.2.2 Proof of Unlinkability

The basic premise of the proof of unlinkability for the protocol presented is that the attacker receives no information about the communication links, except the identity of the nodes belonging to the receiver anonymity set and unlinkability set, while passively observing the execution of the protocol. To back this up we check whether any observable event $o \in O$, the set of all observable events, reveals information that helps to detect communication links. Let $e(p_i)$ be the event 'p_i is the communication path', $e(l_{i,j})$ the event 'there is a communication link between node N_i and N_j', and $e(N_i)$ the event 'N_i is the destination node'.

Theorem 2. *Our protocol guarantees for all observable events $o \in O$ and all communication links $l_{i,j} \in L$ that*

$$\underset{l_{i,j} \in L}{\forall} \underset{o_l \in O}{\forall} I_{E(L),O}(e(l_{i,j}); o_l) = 0,$$

if the attacker already knows the corresponding receiver anonymity set and unlinkability set.

Proof. The following events can be observed by a passive attacker: By observing sending and receiving events, the attacker can infer information on who sends and receives a message, how many messages are sent and received and when the messages are sent and received. Coding information and the size of a message can offer additional information. Traffic patterns, i.e. special characteristics of network traffic, can be captured if the network is observed over a period of time. Using these observations, the attacker tries to reveal communication links. One basic technique is to combine different observations and statistically reveal communication links. In the following, we show the influence of these techniques on the knowledge of the attacker.

We assumed in Section 5.1.4 that the nodes on the path must be selected in such a way, that they are all equally likely to be the destination of the communication. So, for a communication path of p nodes, the probability for all nodes N_i, $1 \leq i \leq p$, to be the destination node N_t is $P(e(N_i)) = \frac{1}{p}$. When the attacker is able to detect who is sending and receiving messages he is able to reduce the unlinkability set to the nodes on the path. We assume that the attacker identifies the source node and tries to find the destination of the communication. For a communication path p_u with p nodes, containing

the nodes $p_u = \{N_1, \cdots, N_p\}$, he can reduce the unlinkability set in the best case to the nodes in p_u. For a network of n nodes, the attacker maximally receives the following amount of information on the destination node N_t of a communication:

$$\begin{aligned} I_{E(N),E(P)}(e(N_t); e(p_u)) &= \log_2 \frac{P(e(N_t)|e(p_u))}{P(e(N_t))} \\ &= \log_2 \frac{\frac{1}{p}}{\frac{1}{n}} \\ &= \log_2 \frac{n}{p}. \end{aligned}$$

Perfect unlinkability can be reached with this protocol if all nodes of the network belong to the communication path. If this is the case the attacker receives no information on the communication link while passively observing the network traffic.

In the following we proof that traffic analysis attacks do not affect the unlinkability offered by the protocol.

Lemma 1. *The protocol for unlinkable communication is secure against packet counting attacks.*

Proof. If the attacker is not able to observe the complete communication the number of sending and receiving events can help to map different nodes to one communication. We showed details of attacks that make use of such observations in Section 4. In the best case, the attacker is able to find all nodes N_i for $1 \leq i \leq p$ that are on the communication path by counting the packets at every node, but is not able to detect the destination node N_t. This is because the packet counts c_i are the same at every node N_i if no other communication interferes the counting, i.e. the packet count is independent of the fact that N_t is the destination of the communication. With $e(c_i)$ as the event 'the attacker observes the packet count c_i' we have:

$$P(e(N_t)|e(c_1), \cdots, e(c_p)) = P(e(N_t)).$$

From this it follows that the mutual information of the identity of N_t and the packet counts c_i, under the assumption that the attacker already knows the nodes of the communication path, is:

$$I_{E(N),E(C)}(e(N_t); e(c_i)) = 0.$$

□

Lemma 2. *The protocol for unlinkable communication is secure against timing attacks.*

Proof. In the same way as for packet counts, the attacker can use the timing of packets in order to map nodes to a communication link. By this method, he is also able to find all nodes on the communication path. During the key-exchange and the forward communication, all nodes perform the same calculations before they forward the message to the next node on the path. The time delay t_i for forwarding a packet on the path is in this case independent of the identity of N_t. During backwards communication, the target node also performs the same computations. The difference is that it also has to prepare the response message. We counter this problem by first sending an announcement to N_t so that it has time to prepare the response in advance. When N_t receives the ticket, the computations are the same as for all other nodes on the path. Let $e(t_i)$ be the event 'the attacker measured the time t_i between the receiving and sending of a message'. When the attacker already knows the communication path, the mutual information of the timings t_i and the identity of N_t is also:

$$I_{E(N),E(T)}(e(N_t); e(t_i)) = 0.$$

□

Lemma 3. *The size, content and the coding of messages sent during the execution of the protocol for unlinkable communication do not reveal any information about the communication link.*

Proof. We have assumed that a fixed packet size s is used for the key-exchange and the communication messages. Therefore, the mutual information of the packet size s_i and the identity of the destination node N_t is 0.

Because of the layered encryption, the attacker has no information on the content of the messages. In the best case, he can distinguish between key-exchange and communication messages, because of the use of asymmetric encryption in the key-exchange message. However, the communication

messages and the key-exchange messages are sent over the complete communication path. As a consequence, these messages offer no information as to whether a node on the path is the destination node or not. For both key-exchange and communication messages, layered encryption prevents an attacker from receiving any information about the content of the messages as this changes at every communication hop. It follows that the attacker is also unable to detect when N_t replaces a ticket message with the response during the reverse communication. Hence, coding of messages offers no information on the communication links. □

Lemma 4. *The protocol for unlinkable communication is secure against communication pattern attacks.*

Proof. Distinctive traffic patterns, i.e. special characteristics of traffic induced by protocols, can be detected even if the communication is encrypted. By using traffic patterns, the attacker can reduce the set of possible destination nodes to the ones that offer the service, that exhibits such a traffic pattern. As we assumed in Section 5.1.4, all nodes on the communication path should offer the same services as N_t, so that the observed traffic patterns match all nodes' services. If the attacker already knows the communication path and all these nodes offer the service, he receives no information on the destination node. □

As result, the attacker receives no information about the communication link by passive observation. □

Our protocol does not prevent the attacker from passively observing the communication and needs no additional dummy traffic to hide the traffic. The basis of unlinkability of this protocol is that, in the best case, all possible observations only allow the identification of the nodes on the communication path. However, they offer no information on which of the nodes on the communication path is the destination node. Therefore, the best passive attacker is able only to reduce the unlinkability set to the p nodes on the path, resulting in a degree of unlinkability of at least $\log_2 p$.

5.3 Practical Considerations

In this section we will show how the protocol can be used in practice. Details on the application of the protocol vary when applying it to different application scenarios. In particular, the method of choosing the communication path can be different because direct communication links between all users of the network do not exist in all scenarios. For example, in ad hoc networks all nodes also act as routers by forwarding messages for other users. As a demonstration, we will adapt the protocol for usage in Internet settings, such as for web browsing or P2P communication, and for ad hoc communication. We built an prototype implementation of the protocol, but refrained from applying it to the scenarios presented in this section. The reason for this is that we wanted to concentrate on the essential functionalities of the protocol. Applying it in these scenarios would introduce problems which are not due to the protocol itself.

5.3.1 Application to Internet Settings

In Internet settings, we can assume that all nodes are pairwise connected, so that it is possible to send messages directly between arbitrary nodes. We can ignore the fact that there are stations on the path between two nodes that belong to the infrastructure of the network.

To apply the protocol to web browsing it is a prerequisite that the target server and additional other servers are able to execute the protocol. If this is the case the sending node N_1 can select randomly the nodes for the communication path from the list of available servers. As was shown in the previous Section 5.2.2, if this is done randomly, the attacker is not able to discover any information by knowing the nodes on the path. The addresses of available servers can be published by means of servers in the Internet, or by using P2P networks.

For the application of the protocol to P2P networks, all nodes that are part of the P2P network execute the protocol and can be part of communication paths. N_1 can choose these nodes randomly in the same way as for web browsing.

5.3.2 Application to Ad Hoc Communication

In ad hoc networks, it is also possible to choose random waypoints and route all messages over these nodes. If there are no direct links between these waypoints, the messages must be routed over several other nodes. As a result, the path over n random nodes consists of totally $n + x$ nodes, where x represents the nodes needed to connect the random waypoints. These x nodes do not belong to the unlinkability set, because a strong attacker might detect that they are not randomly located. Even if the attacker can detect this, the protocol guarantees a degree of unlinkability of $\log_2 n$. However, the more nodes are involved in the routing process, the greater is the cost and the chance of loosing the connection. By minimizing the number of nodes involved we can minimize the cost and maximize the robustness.

For route selection, it is important that all nodes that belong to the unlinkability set have the same likelihood for being the target node. By choosing the cheapest path from the sender node N_1 to the target node N_t we can satisfy this condition. As a consequence, if we assume positive costs for each link, then the cost from N_1 to all other nodes N_i, for $2 \leq i < t$, is optimal. To reach the required size for the unlinkability set, and also to hide the target node on the path, we extend the path to node N_t, so that for all following nodes N_i, $t+1 \leq i \leq t+e$, the resulting path N_1 to N_i is optimal.

If we assume that the network topology is known to all nodes, the sender node N_1 can easily build the cheapest path to the target node N_t by using the Dijkstra algorithm (see [94]). The Dijkstra algorithm initializes the cost of all nodes to ∞ and the cost for N_1 to 0. All nodes are declared as active. In every step it takes the active node N_i with the cheapest cost c_i and path P_i from N_1 to N_i, and checks for all outgoing edges $e(N_i, N_j)$ if $c_i + cost(e(N_i, N_j)) < c_j$. If this is the case it updates the cost c_j to $c_i + cost(e(N_i, N_j))$ and P_j to $P_i \cup N_i$. Additionally, we store for each node N_i the length l_i of the cheapest path in order to know if the desired size for the unlinkability set is reached. After having checked all outgoing edges of N_i we declare N_i as passive. The algorithm is repeated until the target node N_t is the cheapest active node which means that P_t is the cheapest path from N_1 to N_t.

To extend the path only by nodes N_i so that the resulting path from N_1 to N_i is optimal we continue the Dijkstra algorithm in such a way, that we mark all active nodes, whose cheapest path goes over N_t. The algorithm

ends if a marked active node N_i is reached, whose path length l_i is equal to the desired size of the unlinkability set or if there is no further active node marked. In this case we choose the longest path with a marked end node. If the length of this path is l_k, we only have an unlinkability set of l_k nodes. Extending the path by nodes N_i, so that the path from N_1 over N_t to N_i is not optimal, does not increase the unlinkability set if we assume a strong passive attacker who knows how the path is selected.

The limited number of nodes in communication range makes it easy for an attacker to detect all nodes on the communication path. Even in this case, the attacker is not able to detect the communication link. He is only able to reduce the unlinkability set to only those nodes on the communication path. A better degree of unlinkability can be attained if the attacker has no knowledge of the complete communication path. In order to make it harder to detect the path, we can additionally use a technique based on the DC-net. This extension to the basic protocol for unlinkable communication is published in [14, 15]. The main idea is to use DC-net communication with limited group sizes for every one-hop communication step. This technique makes it harder for the attacker to find communication links if he is able only to observe traffic locally. A strong passive attacker is still able to map traffic that belongs to one communication link.

5.4 Conclusion

The presented protocol for unlinkable communication hides the receiver node on a communication path by the use of layered encryption. To keep the computation cost of the protocol small we use efficient symmetric encryption during the communication; less-efficient asymmetric encryption is only used during the key-exchange phase, which is only needed once for every communication path. We introduced a response channel, so that the receiver node is able to send messages back to the sender node without revealing the communication link. We proved that the attacker is not able to obtain any information by passively observing network traffic.

The existing protocols for unlinkable communication presented in Chapter 3 suffer from some weaknesses. The proxy-based approaches with Mix networks and Tor offer good performance. Because the attacker can detect sender and receiver nodes he has a good chance of discovering communica-

tion links even if he has only incomplete knowledge of the network traffic. We eliminate this weakness in our protocol. Receiver nodes are indistinguishable from the nodes that are used to build the unlinkable set, while it produces the same amount of traffic and computational overhead as Tor and Mix networks. In addition to unlinkability of senders and receivers, our protocol also offers receiver anonymity.

The dining cryptographers approach offers provable unlinkability that relies on superposed sending and multicast sending. It is not used in practice because of several weaknesses. One of the main problems of DC-net is its scalability. In order to increase the receiver anonymity set linearly, the message overhead increases quadratically. Our protocol offers comparable receiver anonymity but at considerable lower cost. For both, DC-net and our protocol the message overhead increases only linearly with the unlinkability set.

The expected degree of unlinkability and receiver anonymity provided by our protocol is higher than that for a DC-net. In most network settings, an attacker is not able to observe the complete network traffic, decreasing the chance of observing the complete communication path. In the DC-net scenario, the attacker only needs to observe the traffic of one participant in order to know all nodes of the unlinkability set because all messages are sent to all participants of the DC-net group.

Chapter 6

Conclusion

This chapter summarizes the conclusions and the main research contributions of this thesis. The aim of the thesis was to find a technique that guarantees unlinkable and anonymous communication under the assumption of a strong passive attacker. To reach this goal, we first introduced measures for anonymity and unlinkability. For both measures we used the information theoretic notion of entropy. This technique for measuring anonymity was independently introduced by [9] and [10]. In order to measure unlinkability we defined unlinkability sets, which consist of all possible communication links. We used these measures throughout the thesis for evaluating different approaches to anonymous and unlinkable communication.

We studied the state of the art in the area of anonymous and unlinkable communication in Chapter 3. We called the first group of protocols proxy-based, because their traffic is routed over other stations. The common feature of these approaches is that the communication end points can be detected by an attacker. In Chapter 4 we showed how to exploit this information for traffic analysis attacks. In this kind of attack, the attacker tries to make deductions about the sender and receiver of messages simply by passively observing the network traffic. First, we considered the case where the attacker can infer information by passively observing the network. We then looked at the cases where the attacker actively manipulates traffic or part of the network in order to infer more information. We introduced slotted packet-counting, a new passive attack that improves traditional packet-counting attacks by taking account of the alteration of traffic load (see [12]). By applying this attack to network traffic traces, we were able to demon-

strate how to break existing protocols for anonymity and unlinkability. We showed how the attack can be applied to different scenarios by optimizing its detection rate.

The second group of protocols, those based on the dining cryptographers' network, are not affected by passive attacks because no communication end points are detectable. The reliable broadcast assumption and the poor scalability have the result that the protocol is not used in practice. Additionally, active attackers can disrupt the execution of the protocol, or break the anonymity and unlinkability by chance.

In Chapter 5 we introduced a new protocol for unlinkable communication that also guarantees receiver anonymity. It was published in [13, 14, 11, 15]. The design goal of the protocol was to prevent a passive attacker from obtaining any information through observation of the execution of the protocol. The protocol attains this goal by building communication paths over several nodes, with the destination node placed randomly on the communication path. We were able to prove that by passive observation only, the attacker has no chance of detecting the communication links. Furthermore, we were able to prove that the message overhead of the protocol is optimal in guaranteeing receiver anonymity. We showed means of applying the protocol to different communication settings, such as P2P communication and ad hoc communication.

In the following, we critically review this thesis (Section 6.1). Finally, we discuss open questions which might be interesting for further studies (Section 6.2).

6.1 Discussion and Critical Assessment

In this thesis we assumed a strong passive attacker who is able to observe every communication. Under this assumption most existing protocols for anonymous and unlinkable communication cannot guarantee anonymity and unlinkability. They were designed under the assumption of weaker attacker models, which are more realistic in Internet settings. For example JAP [95] uses Mix cascades to hide the communication link. By distributing these Mix servers geographically it is hard for an attacker to observe a complete communication path. End-to-end mapping of communications is still possible; the distribution merely reduces the chance that an attacker can control

or observe a sufficient number of servers to provide information about a complete communication path. Previous knowledge can make attacks on these protocols easier, e.g. initial suspicion that a particular communication link is being used.

Most of the attacks presented can theoretically break the anonymity and unlinkability of most protocols. Already some of these attacks have been successfully applied in practice against protocols such as Tor. For most users it is sufficient if no direct linking of sender and receiver is possible because there is often no direct damage when unattributable communication is detected. The cost of applying these attacks is sufficiently great so that only few institutions are interested in performing them. On the other hand, the loss in performance due to our protocol are similar to that of Tor. The difference is that, when our protocol is used no attacker can break unlinkability by using passive observation.

We proved for our protocol presented in Chapter 5 that a passive attacker is able to detect the unlinkability set, while only observing the traffic. We have already mentioned that the attacker still might be able to detect communication patterns even if the communication is encrypted. We assumed that all nodes offer the same service, so that communication patterns might be due to any of several different communication links. If it is not possible to choose the nodes on the path in this way, additional dummy traffic can hide communication patterns. Wright et al. [96] address this problem by introducing traffic morphing where a class of traffic can be morphed to another class of traffic. The optimal insertion method for dummy traffic might be an interesting topic for further studies.

More generally, previous knowledge can give the attacker information that helps to break anonymity and unlinkability. For example, if an attacker knows that messages are sent more frequently to some nodes, he might be able to decrease the degree of anonymity and unlinkability. This previous knowledge is independent of the protocol used, because it is collected before the use of the protocol. Further studies on how to prevent or detect previous knowledge might be interesting.

Another problem is that every communication partner has to execute the protocol. In some settings this is easy to implement, for example in P2P networks, where every peer performs the same action and has similar interests. In contrast, this is not the case for web browsing. In order to hide

a users' contact with a special server R, which hosts a web site, a communication path over different web servers including R must be constructed. This requires that R and all servers on the communication path are able to execute the protocol. Operators of servers have no incentive for providing unlinkable communication. Indeed, the opposite is the case: most operators of servers are interested in identifying their users. For those scenarios, incentives for executing the protocol have to be introduced. To do this, trust and cooperation techniques can be applied. Further studies in this area are needed.

We showed in Chapter 5 that the protocol offers receiver anonymity at minimal cost. For this proof we assumed the strong passive attacker model. In real-world settings, where an attacker has only partial knowledge of the network, the degree of receiver anonymity is even higher.

6.2 Further Work

There are many fields for further investigation in the area of unlinkability and anonymous communication. In relation to the protocol presented, it is interesting to study the influence of previous knowledge, i.e. knowledge that is not revealed during the execution of the protocol but nevertheless decreases the degree of unlinkability or receiver anonymity. For example, if some nodes have a higher probability of being the end point of a communication, the degrees of unlinkability and receiver anonymity decrease even if the protocol leaks no additional information. The question here is whether previous knowledge can be detected, e.g. with Trust techniques, and whether it is possible to increase the degree of unlinkability and receiver anonymity in these cases by applying additional techniques. One way could be to insert additional dummy traffic in order to decrease previous knowledge. Applying the measures for unlinkability and receiver anonymity can help to find nodes in the network, where injecting directed dummy traffic could increase the entropy with minimal cost. Such a technique would be also applicable to other protocols.

In the presented protocol, the attacker is not able to obtain any information by passive observation. If he controls some nodes of the communication path he might be able to reduce the degree of unlinkability and receiver anonymity. To detect these malicious nodes, trust techniques might be ap-

plicable. Trust techniques are based on the idea of collecting information on a subject of interest in order to estimate whether he is honest or malicious. In order to apply it for our needs, the main task lies in the detection of malicious behavior. This is a general problem for trust applications. In our case the unlinkable and anonymous communication additionally complicates the acquisition of information.

As we mentioned in the previous Section 6.1, one problem is that not all nodes might be interested in participating in the unlinkability set. For example, when using the protocol for web surfing, the web servers have no incentives for executing the protocol. One way to solve this problem would be by using virtual currencies (see e.g. [97]), where the initiator of a communication pays to send messages anonymously. In P2P networks similar techniques can be applied in order to prevent free-riders from using the protocol without contributing to the unlinkability set for others' communications (see e.g. [98]). Further studies in applying these techniques for use in combination with the protocol should be undertaken.

In this thesis, we have assumed a strong passive attacker who is able to observe every communication in the network. The bounds we found during our theoretical analysis for the degree of anonymity and unlinkability are based on this attacker model. Weaker attackers might also be able to break the anonymity and unlinkability of some protocols but in most cases the degrees of anonymity and unlinkability increase as the attacker becomes weaker. In our work we concentrated on the lower bound for anonymity and unlinkability that is guaranteed by the protocol in the face of the most powerful passive attacker possible. Further investigations could be made in order to specify the expected degree of anonymity and unlinkability when assuming more realistic attacker models.

Bibliography

[1] "The internet archive," February 2009. http://www.archive.org.

[2] D. Presse-Agentur, "Ermittlungen wegen Daten-Diebstahl bei T-Mobile vor Wiederaufnahme," December 2008. http://www.heise.de/newsticker/meldung/117035.

[3] "Steuer-Skandal - Informant enttarnt," February 2008. http://www.n-tv.de/921125.html.

[4] T. Online, "UK citizens' private information being lost at record rate," February 2009. http://www.timesonline.co.uk/tol/news/politics/article5688347.ece.

[5] The Council of Europe, "Convention no. 108, Convention for the protection of individuals with regard to automatic processing of personal data," 1981.

[6] EU Commission, "Directive 2006/24/ec of the European Parliament and of the Council of 15 march 2006," *Official Journal of the European Union*, vol. L 105/54, 2006.

[7] T. Online, "Taxman loses sensitive personal data on 25m people," November 2007. http://www.timesonline.co.uk/tol/news/uk/article2907495.ece.

[8] J. Kuri, "Kriminalbeamte fordern zentrale Datenbank für Verbindungsdaten," December 2008. http://www.heise.de/newsticker/meldung/108497.

[9] A. Serjantov and G. Danezis, "Towards an information theoretic metric for anonymity," in *Proceedings of Privacy Enhancing Technologies*

Workshop (PET 2002) (R. Dingledine and P. Syverson, eds.), Springer-Verlag, LNCS 2482, April 2002.

[10] C. Díaz, S. Seys, J. Claessens, and B. Preneel, "Towards measuring anonymity," in *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*, April 2002.

[11] V. Fusenig, E. Staab, U. Sorger, and T. Engel, "Unlinkable communication," in *Proceedings of the Sixth Annual Conference on Privacy, Security and Trust (PST2008)*, (Fredericton, New Brunswick, Canada), pp. 51–55, October 1-3 2008.

[12] V. Fusenig, E. Staab, U. Sorger, and T. Engel, "Slotted packet counting attacks on anonymity protocols," in *Proceedings of the 7th Australasian Information Security Conference (AISC 2009)* (L. Brankovic and W. Susilo, eds.), vol. 98 of *CRPIT*, (Wellington, New Zealand), ACS, 2009.

[13] V. Fusenig, D. Spiewak, and T. Engel, "Acimn protocol: A protocol for anonymous communication in multi hop wireless networks," in *IEEE Wireless Telecommunications Symposium (WTS 2007), April 26-28 2007*, IEEE, IEEE, April 2007.

[14] V. Fusenig, D. Spiewak, and T. Engel, "Acimn: A protocol for anonymous communication in multi hop wireless networks," in *Proceedings of the Sixth Australasian Information Security Conference (AISC 2008)* (L. Brankovic and M. Miller, eds.), vol. 81 of *CRPIT*, (Wollongong, NSW, Australia), pp. 107–114, ACS, 2008.

[15] V. Fusenig, D. Spiewak, and T. Engel, "Anonymous communication in multi hop wireless networks," *Journal of Research and Practice in Information Technology*, vol. 40, no. 3, pp. 205–224, 2008.

[16] M. Rehak, E. Staab, V. Fusenig, J. Stiborek, M. Grill, K. Bartos, M. Pechoucek, and T. Engel, "Threat-model-driven runtime adaptation and evaluation of intrusion detection system," in *Proceedings of the 6th International Conference on Autonomic Computing and Communications*, (Barcelona, Spain), June 2009.

[17] E. Staab, V. Fusenig, and T. Engel, "Towards trust-based acquisition of unverifiable information," in *Cooperative Information Agents XII* (M. Klusch, M. Pechoucek, and A. Polleres, eds.), vol. 5180 of *LNCS*, pp. 41–54, Springer Verlag, 2008.

[18] E. Staab, V. Fusenig, and T. Engel, "Trust-aided acquisition of unverifiable information," in *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI '08)*, pp. 869–870, IOS Press, 2008.

[19] D. Spiewak, V. Fusenig, and T. Engel, "The importance of location on trust in mobile networks," *WESEAS Transactions on Communication*, vol. 7, no. 5, pp. 349–360, 2008.

[20] D. Spiewak, V. Fusenig, and T. Engel, "Trustrings in mobile wireless network settings," *Proceedings of the INFORMATION SECURITY and PRIVACY Conference 2007, Puerto de la Cruz, Spain*, vol. 6, pp. 104–110, December 2007.

[21] D. Spiewak, V. Fusenig, and T. Engel, "Mobility diversifies trust: Introducing trustrings," in *IEEE Wireless Telecommunications Symposium (WTS 2007), April 26-28 2007*, IEEE, IEEE, April 2007.

[22] D. Spiewak, T. Engel, and V. Fusenig, "Unmasking threats in mobile wireless ad-hoc network settings," *WSEAS Transactions on Communications, Issue 1*, vol. 6, pp. 104–110, January 2007.

[23] D. Spiewak, T. Engel, and V. Fusenig, "Towards a threat model for mobile ad-hoc networks," in *Proceedings of the 5th Int.Conf. on INFORMATION SECURITY and PRIVACY "'ISP'06"', November 20-22, 2006, Venice, Italy*, vol. 5, WSEAS, ISS, November 2006.

[24] U. Roth and V. Fusenig, "How certain is recommended trust-information," in *Proceedings of the 15th International World Wide Web Conference (WWW2006), Workshop Models of Trust for the Web (MTW'06), May 22-26, 2006, Edinburgh, Scotland*, May 2006.

[25] U. Roth and V. Fusenig, "Trust-decisions on the base of maximal information of recommended direct-trust," in *Proceedings of the 2006 ACM Symposium on Applied Computing 2006, ACM SAC 2006, 23-27. April 2006, Dijon, France*, vol. 2, pp. 1898–1899, ACM, ACM, April 2006.

[26] H. Bauer, *Maß- und Integrationstheorie*. Gruyter, 1992.

[27] R. Johannesson, *Informationstheorie. Grundlage der (Tele-) Kommunikation*. Addison Wesley Verlag, 1992.

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

[29] The Century Dictionary, *The Century Dictionary*. The Century Co., 1911.

[30] A. Pfitzmann and M. Köhntopp, "Anonymity, unobservability, and pseudonymity - a proposal for terminology," in *Workshop on Design Issues in Anonymity and Unobservability* (H. Federrath, ed.), vol. 2009 of *Lecture Notes in Computer Science*, pp. 1–9, Springer, 2000.

[31] M. Autenrieth, "Unverzichtbarer Schutz gegen existenzielles Risiko," May 2008. `http://www.sueddeutsche.de/finanzen/767/441508/text/`.

[32] C. E. Shannon, "Communication theory of secrecy systems," *Bell System Technical Journal*, vol. 28(4), pp. 656–715, 1948.

[33] A. Kerckhoffs, "La cryptographie militaire," *Journal des Sciences Militaires*, vol. IX, pp. 5–38, January 1883.

[34] A. Serjantov and P. Sewell, "Passive attack analysis for connection-based anonymity systems," in *Proceedings of ESORICS 2003*, October 2003.

[35] D. Dolev and A. C. Yao, "On the security of public key protocols," tech. rep., Stanford University, Stanford, CA, USA, 1981.

[36] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[37] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 4, February 1981.

[38] D. Chaum, "The dining cryptographers problem: Unconditional sender and recipient untraceability," *Journal of Cryptology*, vol. 1, pp. 65–75, 1988.

[39] A. Pfitzmann, A. Juschka, A.-K. Stange, S. Steinbrecher, and S. Köpsell, "Communication privacy," in *Digital Privacy: Theory, Technologies and Practices*, Elsevier, 2007.

[40] A. Jerichow, J. Müller, A. Pfitzmann, B. Pfitzmann, and M. Waidner, "Real-Time MIXes: A Bandwidth-Efficient Anonymity Protocol," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, 1998.

[41] C. Díaz and A. Serjantov, "Generalising mixes," in *Proceedings of Privacy Enhancing Technologies workshop (PET 2003)*, pp. 18–31, March 2003.

[42] G. Danezis, R. Dingledine, D. Hopwood, and N. Mathewson, "Mixminion: Design of a type III anonymous remailer protocol," in *In Proceedings of the 2003 IEEE Symposium on Security and Privacy*, pp. 2–15, 2003.

[43] O. Berthold, H. Federrath, and S. Köpsell, "Web mixes: a system for anonymous and unobservable internet access," in *International workshop on Designing privacy enhancing technologies*, (New York, NY, USA), pp. 115–129, Springer-Verlag New York, Inc., 2001.

[44] D. M. Goldschlag, M. G. Reed, and P. F. Syverson, "Hiding Routing Information," in *Proceedings of Information Hiding: First International Workshop* (R. Anderson, ed.), pp. 137–150, Springer-Verlag, LNCS 1174, May 1996.

[45] P. F. Syverson, D. M. Goldschlag, and M. G. Reed, "Anonymous connections and onion routing," in *SP '97: Proceedings of the 1997 IEEE Symposium on Security and Privacy*, (Washington, DC, USA), p. 44, IEEE Computer Society, 1997.

[46] P. F. Syverson, M. G. Reed, and D. M. Goldschlag, "Private web browsing," *J. Comput. Secur.*, vol. 5, no. 3, pp. 237–248, 1997.

[47] P. F. Syverson, M. G. Reed, and D. M. Goldschlag, "Onion Routing access configurations," in *DARPA Information Survivability Conference and Exposition (DISCEX 2000)*, vol. 1, pp. 34–40, IEEE CS Press, 2000.

[48] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: the second-generation onion router," in *SSYM'04: Proceedings of the 13th conference on USENIX Security Symposium*, (Berkeley, CA, USA), pp. 21–21, USENIX Association, August 2004.

[49] K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker, "Low-resource routing attacks against Tor," in *WPES '07: Proceedings of the 2007 ACM workshop on Privacy in electronic society*, (New York, NY, USA), pp. 11–20, ACM, 2007.

[50] M. K. Wright, M. Adler, B. N. Levine, and C. Shields, "The predecessor attack: An analysis of a threat to anonymous communications systems," *ACM Trans. Inf. Syst. Secur.*, vol. 7, no. 4, pp. 489–522, 2004.

[51] J. Bos and B. den Boer, "Detection of disrupters in the DC protocol," in *EUROCRYPT '89: Proceedings of the workshop on the theory and application of cryptographic techniques on Advances in cryptology*, (New York, NY, USA), pp. 320–327, Springer-Verlag New York, Inc., 1990.

[52] A. Pfitzmann, "How to implement ISDNs without user observability - some remarks," *SIGSAC Rev.*, vol. 5, no. 1, pp. 19–21, 1987.

[53] M. Waidner, "Unconditional sender and recipient untraceability in spite of active attacks," in *EUROCRYPT '89: Proceedings of the workshop on the theory and application of cryptographic techniques on Advances in cryptology*, (New York, NY, USA), pp. 302–319, Springer-Verlag New York, Inc., 1990.

[54] C. Studholme and I. Blake, "Multiparty computation to generate secret permutations." Cryptology ePrint Archive, Report 2007/353, 2007.

[55] M. Waidner and B. Pfitzmann, "The dining cryptographers in the disco: unconditional sender and recipient untraceability with computationally secure serviceability," in *EUROCRYPT '89: Proceedings of the workshop on the theory and application of cryptographic techniques on Advances in cryptology*, (New York, NY, USA), p. 690, Springer-Verlag New York, Inc., 1990.

[56] S. Goel, M. Robson, M. Polte, and E. G. Sirer, "Herbivore: A Scalable and Efficient Protocol for Anonymous Communication," Tech. Rep. 2003-1890, Cornell University, Ithaca, NY, February 2003.

[57] J. Kong and X. Hong, "ANODR: anonymous on-demand routing with untraceable routes for mobile ad-hoc networks," in *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, (New York, NY, USA), pp. 291–302, ACM Press, 2003.

[58] O. Berthold, A. Pfitzmann, and R. Standtke, "The disadvantages of free mix routes and how to overcome them," in *International workshop on Designing privacy enhancing technologies*, (New York, NY, USA), pp. 30–45, Springer-Verlag New York, Inc., 2001.

[59] J.-F. Raymond, "Traffic analysis: protocols, attacks, design issues, and open problems," in *Designing Privacy Enhancing Technologies: Proceedings of International Workshop on Design Issues in Anonymity and Unobservability* (H. Federrath, ed.), vol. 2009 of *LNCS*, (New York, NY, USA), pp. 10–29, Springer-Verlag New York, Inc., 2001.

[60] S. M. Huffmann and M. H. Reifer, "Method for geolocating logical network addresses." Patent of the United States of America as represented by the Director, National Security Agency, September 2005. Patent Number: 6,947,978.

[61] M. Dusi, M. Crotti, F. Gringoli, and L. Salgarelli, "Tunnel hunter: Detecting application-layer tunnels with statistical fingerprinting," *Comput. Netw.*, vol. 53, no. 1, pp. 81–97, 2009.

[62] D. Agrawal and D. Kesdogan, "Measuring anonymity: The disclosure attack," *IEEE Security & Privacy*, vol. 1, no. 6, pp. 27–34, 2003.

[63] D. Kesdogan, D. Agrawal, and S. Penz, "Limits of anonymity in open environments," in *IH '02: Revised Papers from the 5th International Workshop on Information Hiding*, (London, UK), pp. 53–69, Springer-Verlag, 2003.

[64] S. Penz, "Security analysis and evaluation of anonymity techniques in open environments," Master's thesis, Computer Science Department Informatik IV (Communication and Distributed Systems), Technical University of Aachen, 2002.

[65] D. Agrawal, D. Kesdogan, and S. Penz, "Probabilistic treatment of mixes to hamper traffic analysis," in *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy*, (Washington, DC, USA), p. 16, IEEE Computer Society, 2003.

[66] G. Danezis, "Statistical disclosure attacks: Traffic confirmation in open environments," in *Proceedings of Security and Privacy in the Age of Uncertainty, (SEC2003)*, (Athens), pp. 421–426, IFIP TC11, May 2003.

[67] D. Kesdogan and L. Pimenidis, "The hitting set attack on anonymity protocols," in *Information Hiding* (J. J. Fridrich, ed.), vol. 3200 of *Lecture Notes in Computer Science*, pp. 326–339, Springer, 2004.

[68] N. Borisov, G. Danezis, P. Mittal, and P. Tabriz, "Denial of service or denial of security?," in *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, (New York, NY, USA), pp. 92–102, ACM, 2007.

[69] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2." RFC 5246 (Proposed Standard), Aug. 2008.

[70] A. Back, U. Möller, and A. Stiglic, "Traffic analysis attacks and trade-offs in anonymity providing systems," in *IHW '01: Proceedings of the 4th International Workshop on Information Hiding*, (London, UK), pp. 245–257, Springer-Verlag, 2001.

[71] C. Gulcu and G. Tsudik, "Mixing email with Babel," in *SNDSS '96: Proceedings of the 1996 Symposium on Network and Distributed System Security (SNDSS '96)*, (Washington, DC, USA), p. 2, IEEE Computer Society, 1996.

[72] A. Serjantov, R. Dingledine, and P. Syverson, "From a trickle to a flood: Active attacks on several mix types," in *Proceedings of Information Hiding Workshop (IH 2002)* (F. Petitcolas, ed.), Springer-Verlag, LNCS 2578, October 2002.

[73] G. Danezis and L. Sassaman, "Heartbeat traffic to counter (n-1) attacks: red-green-black mixes," in *WPES '03: Proceedings of the 2003 ACM workshop on Privacy in the electronic society*, (New York, NY, USA), pp. 89–93, ACM, 2003.

[74] X. Wang and D. S. Reeves, "Robust correlation of encrypted attack traffic through stepping stones by manipulation of interpacket delays," in *ACM Conference on Computer and Communications Security*, pp. 20–29, 2003.

[75] X. Wang, S. Chen, and S. Jajodia, "Network flow watermarking attack on low-latency anonymous communication systems," in *IEEE Symposium on Security and Privacy*, pp. 116–130, 2007.

[76] Y. J. Pyun, Y. H. Park, X. Wang, D. S. Reeves, and P. Ning, "Tracing traffic through intermediate hosts that repacketize flows," in *INFOCOM*, pp. 634–642, IEEE, 2007.

[77] N. Kiyavash, A. Houmansadr, and N. Borisov, "Multi-flow attacks against network flow watermarking schemes," in *USENIX Security Symposium*, pp. 307–320, USENIX, 2008.

[78] W. Dai, "Two attacks against freedom," December 2008. `http://www.weidai.com/freedom-attacks.txt`.

[79] P. Golle and A. Juels, "Dining cryptographers revisited," in *EUROCRYPT* (C. Cachin and J. Camenisch, eds.), vol. 3027 of *Lecture Notes in Computer Science*, pp. 456–473, Springer, 2004.

[80] C. Franck, "New directions for dining cryptographers," Master's thesis, University of Luxembourg, 2008.

[81] L. J. Bain and M. Engelhardt, *Introduction to Probability and Mathematical Statistics*. Duxbury Press, 2nd, ed., 2000.

[82] B. N. Levine, M. K. Reiter, C. Wang, and M. K. Wright, "Timing attacks in low-latency mix-based systems," in *Proceedings of Financial Cryptography (FC '04)*, pp. 251–265, February 2004.

[83] V. Paxson and S. Floyd, "Wide-area traffic: the failure of Poisson modeling," *SIGCOMM Comput. Commun. Rev.*, vol. 24, no. 4, pp. 257–268, 1994.

[84] C. J. V. Rijsbergen, *Information Retrieval*. Newton, MA, USA: Butterworth-Heinemann, 1979.

[85] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "MASK: Anonymous on-demand routing in mobile ad hoc networks," in *Transactions on Wireless Communications*, vol. 21, pp. 2376–2385, IEEE, 2006.

[86] S. Seys and B. Preneel, "ARM: Anonymous routing protocol for mobile ad hoc networks," in *Proceedings of the 20th IEEE International Conference on Advanced Information Networking and Applications - Workshops (AINA 2006 Workshops)*, (Vienna,AU), pp. 133–137, IEEE, 2006.

[87] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644–654, 1976.

[88] P. R. Zimmermann, *The official PGP user's guide*. Cambridge, MA, USA: MIT Press, 1995.

[89] V. Rijmen and J. Daemen, *The Design of Rijndael: AES. The Advanced Encryption Standard*. Springer, Berlin, 2002.

[90] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 2001.

[91] Unitymedia, "Internet, Telefon und TV," March 2009. `http://www.unitymedia.de/`.

[92] IEEE, *IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements; Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. IEEE Computer Society, June), year = 2007.

[93] T. Scherer and T. Engel, "Bandwidth consumption for providing fair internet access in wireless mesh networks," in *The 2006 IEEE International Workshop on Wireless Ad-hoc and Sensor Networks (IWWAN 2006)*, June 2006.

[94] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[95] "JAP Anon Proxy," February 2009. `http://anon.inf.tu-dresden.de/`.

[96] C. Wright, S. Coull, and F. Monrose, "Traffic morphing: An efficient defense against statistical traffic analysis," in *Proceedings of the Network and Distributed Security Symposium - NDSS '09*, IEEE, February 2009.

[97] L. Buttyán and J.-P. Hubaux, "Stimulating cooperation in self-organizing mobile ad hoc networks," *Mob. Netw. Appl.*, vol. 8, no. 5, pp. 579–592, 2003.

[98] M. Feldman and J. Chuang, "Overcoming free-riding behavior in peer-to-peer systems," *SIGecom Exch.*, vol. 5, no. 4, pp. 41–50, 2005.