Formalizing Anonymity: A Review

University of York Technical Report YCS 389

Joss Wright, Susan Stepney, John A. Clark, Jeremy Jacob
{joss,susan,jac,jeremy.jacob}@cs.york.ac.uk

21st June 2005

Abstract

This review provides an overview of the current state of the field of anonymity research with particular reference to the formal specification of anonymity as a security property.

Contents

Chapter 1

Introduction

In order to avoid discrimination against, or abuse of, users of communicating systems it may sometimes be necessary to prevent their activities from being made public. In many cases this may be achieved by robust encryption protocols for transactions, however in a number of important cases it is necessary for the content of transactions to be publicly available.

In other situations the very use of a particular system may be incriminating. Certain websites may be censored, or simply viewed unfavourably by parties with access to web browsing logs. The use of peer-to-peer file sharing networks, while legal in itself, can cause suspicion due to the high volume of illegal content. In these situations, the legitimate user may well desire privacy to prevent the assumption of guilt that others may associate with these uses.

Alternatively, a user may simply not wish their participation in an online system to be noted or traceable. Online discussion mailing lists and newsgroups are frequently archived in publicly accessible locations. The senders of spam email are known to harvest these locations for email addresses to populate spam databases. Users taking part in such lists may wish to avoid having their participation logged to avoid an increase of unsolicited email to their accounts.

In all of these cases, a useful and effective method for protection of the user is to sever the link between a user's identity and their observable behaviour in the system. In short: to make the user anonymous.

Several methods to achieve this goal have been proposed and, at least partially, implemented. Yet, despite some formal foundations in the literature, there has been very little rigorous design or verification work in the systems that have emerged. Indeed, there have been almost no proposed formal specifications of anonymity properties that are of utility in modelling anything more than academic or toy systems. As a result of this, nearly every system that has been released has imperfectly protected the identity of its users.

This review focuses on identity hiding, usually characterized by the linked

properties of anonymity and pseudonymity, with a view to their formal specification and quantification. In addition, attention is given to related security properties that accompany identity hiding in usable systems, including reputation, privacy and fairness.

Chapter 2

Background and Terminology

> "We're an information economy. They teach you that in school. What they don't tell you is that it's impossible to move, to live, to operate at any level without leaving traces, bits, seemingly meaningless fragments of personal information. Fragments that can be retrieved, amplified..."
>
> – William Gibson, *"Johnny Mnemonic"* (1981)

We first discuss some terms that provide a foundation on which to build more formal descriptions. Work in this section draws on many separate papers in the field, however special mention is due to Pfitzmann and Köhntopp (2000).

2.1 Privacy

> "Nec vixit male qui natus moriensque fefellit."
> *(Not a bad life is that of one who is born and dies in obscurity)*[1]
>
> – Horace, *Epistles, I, 17, 10* (c.20 BCE)

The desire for privacy motivates much of the research into anonymity systems. Anonymity provides a mechanism by which an individual may render their actions free from observation, and thus protect the privacy of their actions.

The notion of privacy has been discussed at least as far back as Aristotle (384–327 BCE), who defined the separate spheres of public life: '$\pi o \lambda \iota \varsigma$' (*polis*, city) and private life '$o \iota \kappa o \varsigma$' (*oikos*, home). The importance of these concepts is illustrated

[1] This quote, often found on the Web attributed to Cicero, is normally seen translated as "Nor has he spent his life badly who has passed it in privacy." The translation shown here is our own.

by the derivation from them of the English words *politics* and *economics*. Undoubtedly, Aristotle was not the first to elucidate these ideas.

Within Britain, privacy has arguably been part of the law since 1361 with the adoption of the Justices of the Peace Act under the reign of Edward III. This act included punishments for "peeping toms" and eavesdroppers, themselves derogatory terms that reflect the already negative view held by society towards those who invade the privacy of others.

Despite this early recognition of personal privacy, privacy as an independent right has been explicitly recognized only in the past 150 years. One of the first extant definitions was made by the United States Supreme Court Justice Louis Brandeis and lawyer Samuel Warren (Brandeis and Warren, 1890), who examined the right to privacy as a natural extension of the individual right to liberty. They stated that "liberty" as a right had initially been enforced with respect to preventing physical assault, but that as newer business models and media coverage started to have significant effects in society, intrusion into private lives for public consumption became of concern to many, and the ideal of liberty was necessarily extended to include unfair intervention into aspects of a person's life that could be embarrassing or dangerous if publicized:

> "Gossip is no longer the resource of the idle and of the vicious, but has become a trade, which is pursued with industry as well as effrontery... The intensity and complexity of life, attendant upon advancing civilization, have rendered necessary some retreat from the world, and man, under the refining influence of culture, has become more sensitive to publicity, so that solitude and privacy have become more essential to the individual; but modern enterprise and invention have, through invasions upon his privacy, subjected him to mental pain and distress, far greater than could be inflicted by mere bodily injury." (Brandeis and Warren, 1890)

In reference to earlier work by a Michigan Supreme Court Justice (Cooley, 1888), Brandeis and Warren define privacy as "the right to be let alone". This concept is still fundamental to almost all definitions of personal privacy.

Real interest in privacy, however, appears to have begun only in the second half of the twentieth century. The United Nations Universal Declaration of Human Rights embodies the right to privacy in its twelfth article:

> "No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honour and reputation. Everyone has the right to the protection of the law against such interference or attacks." (United Nations, 1948)

Similarly, the International Covenant on Civil and Political Rights (ICCPR, 1997), a section of the International Bill of Rights adopted to expand on and clar-

ify the Universal Declaration of Human Rights, reiterates the fundamentality of privacy as a right to which humans are entitled.

Both the European Union and the United States have measures to protect privacy (European Union, 1998; United States Department of Commerce, 2004), however these rights are still emerging and in a state of constant alteration.

In British law, the first explicit electronic privacy legislation was the 1984 Data Protection Act (HMSO, 1984). This act, superseded by the 1998 Act of the same name, regulates the uses to which data processed by automatic equipment may be put. The legislation places strict limitations on individuals who store the personal data of third parties, requiring that any information is subject to a number of "data protection principles". These principles include the right of the subject to inspect any stored data, and the duty of the holder of the information to provide adequate technical protection for such data. The holder of the data is also obliged to ensure that the data is not used for purposes other than those for which the data was originally collected, unless such a use has been agreed to by the data subject.

The British 1998 Data Protection Act (HMSO, 1998) implements the 1998 European Union Directive on Data Protection (European Union, 1998). This EU directive, enforced across the EU's economic sphere, includes the principle that personal data must not pass to countries outside the European Union that do not also implement adequate data protection measures as defined by the directive. This has caused some difficulties with the passage of data to the United States, which relies on corporate self-regulation (United States Department of Commerce, 2004) rather than legislation to protect privacy, and thus does not meet the requirements of the EU directive. Various programmes are in place to overcome this barrier between two of the world's major economic forces, however the insistence of the EU on strict controls over such a valuable resource as personal data is pushing the world at large towards strong data protection policies.

There have been analyses of privacy that do not support its importance in society. Most notably, the work of Posner (1981) examines privacy from an economic basis. This view states that personal information should be kept private only if the economic value to society of such information is decreased by its becoming public knowledge. Posner argues that the only personal value in concealing private information is in deceiving or manipulating others for personal gain, and thus is not of economic use to society as a whole. This view proposes corporate privacy as having value, but asserts that personal privacy is not beneficial to a nation's economy and so should not be protected in law.

Posner's view of privacy has not achieved wide acceptance, and many societies in the modern world have enacted laws that protect privacy to a greater or lesser extent. The increasing level of information that is transmitted, and thus may be stored, by computers is causing a greater public interest in the right to privacy. In the past, privacy was naturally protected by the difficulty of storing and collating

personal data. As surveillance and storage become increasingly feasible, it has become necessary to protect this right in more detailed and effective ways.

The privacy legislation and regulations being enacted in various countries across the world are a useful step in preventing the widespread storage and dissemination of personal information. There are, however, always parties that seek to bypass these measures, for a variety of reasons. Rather than regulate the storage and transmission of personal information, advocates of anonymity propose to achieve the same goals by preventing such information ever becoming available. This position provides the motivation for the research covered in this review.

2.2 Names and Identity

Many of the concepts examined in this review are in some way concerned with the obfuscation of information that relates to a user's identity. This information can take many forms, but the classic example is the *name*. The name of an individual is intended to be a unique identifier within some group that allows for that individual to be distinguished from the other members of that group. The fact that, in the increasingly large social groups in which we find ourselves, a classical name is rarely unique does not affect the purpose of distinguishing those around us by such labels.

When we discuss the anonymity properties of an agent, we are implicitly assuming definitions of identity. We assume that a user of a system is a unique individual who performs actions that can potentially be traced by another individual. However, there is a dissociation between a user's representation in a system and their "real-world" persona. Multiple users can collaborate to form a single online identity, and a single user may have multiple representations online. The implications of this are not fully understood, as the simplifying assumption that a single user is linked to a single representation is almost uniformly made in the field.

While it is possible to work with definitions of a "name" that rely on our implicit assumptions concerning identity, it should be remembered that there are many possible interpretations of what constitutes an identity. In forming a model of anonymity properties, the strict definition of these properties must play an important rôle.

2.3 Anonymity

The English word *anonymous* is derived from the classical Greek stem 'ονιμα' (*onyma*, name), combined with the prefix α- (*a*-, the absence or lack of a prop-

erty). "Anonymity" may therefore be understood as the state of being nameless, or having an absence of identification.

To extend this definition to more common use within the field, consider

> "Anonymity is the state of being not identifiable within a set of subjects"
> Pfitzmann and Köhntopp (2000)

This definition succinctly expresses the main purpose of anonymity, although it leaves questions open concerning what is meant by the term "identifiable".

Anonymity is the fundamental identity hiding property, providing total removal of identifying information from its subject. Any identifying information required by users or computers can be added into a data channel within an anonymous system. As such, anonymity provides us with the choice to limit identity hiding as much or as little as desired by explicitly revealing identifying information when it is necessary.

Total anonymity is the focal point for research into identity hiding. Additionally, anonymity systems are uniformly based on a small set of possible approaches, Chaum's *mix* (Chaum, 1981) being the most significant of these. (These approaches are treated in detail in sections 3 and 4). The most active topic of research into identity hiding is therefore the finer details of the various subtle variations of these basic ideas.

Despite this focus on anonymous systems, total anonymity is very much a two-edged sword. For certain forms of application, such as posting to mailing lists or accessing the world wide web, anonymity can be a highly desirable goal. Other systems, however, suffer greatly if there is no possibility of tracking identities. Sometimes identity needs to be tracked over the course of an extended transaction, but not between transactions. For this reason *pseudonymous communication*, which provides a certain amount of information associated with an identity, is required for a number of practical identity hiding systems.

2.3.1 Sender and Recipient Anonymity

We have already defined anonymity as the property of being nameless, given certain assumptions concerning the meaning of a "name". However, this namelessness is only of use or interest in a communicating system. Communication requires two participants: a sender and a recipient, where either participant may in actuality be a group of individuals. To whom does this anonymity apply?

This question divides our definition of anonymity into three. We may ensure the anonymity of the initiator of some communication, but leave the recipient's identity open to the world. Conversely it is possible for a sender to make available their identity, but to ensure that the recipient of their message remains unknown. This provides a very different problem to sender anonymity, and is also

less studied. Finally, it may be desirable for both end-points of communication to be hidden from observation. Each of these forms of anonymity has its own set of applications, design problems and potential attacks.

2.3.2 Connection-Based and Message-Based Anonymity

Another important distinction is the form of the communication that takes place between the sender and the recipient. The majority of data transmission over the Internet uses the connection-based Transmission Control Protocol (Postel, 1981). However at a higher level, much of the actual communication that takes place, especially email, is message-based.

Current literature focuses mainly on message-based anonymity, which is an easier problem than that of anonymity within connection-based systems. There are some applications, such as remote terminal sessions, that simply cannot be performed effectively using a message-based approach. For these systems, it is necessary to use a connection-based approach.

A number of approaches to connection-based anonymity have been proposed. By far the most popular of these is "onion routing" (Goldschlag et al., 1996). The main concern with such systems is the requirement for low-latency connections and bandwidth restrictions. Message-based systems inherently cause delays, unacceptable in connection-based systems. This makes it necessary to use other approaches that avoid these concerns, but which then open up a new set of possible attacks.

2.3.3 The Anonymity Set

The traditional method of quantifying anonymity is to use the size of the "anonymity set": the set of all participants who could have performed an action. This approach was proposed by Chaum (1981), where it was used to analyse the anonymity provided by a mix (see section 3). The larger the size of the set of participants that could have performed an action, the stronger the anonymity provided by the system.

This quantification, while certainly of use, is not ideal. The most critical of the deficiencies of the anonymity set is that it assumes a uniform distribution of probabilities across the set of participants. This assumption is rather naïve for a group of heterogeneous users. In response to this, a number of alternative quantification methods have been proposed that seek to deal with both this and other problems inherent in the anonymity set (Serjantov and Danezis, 2002). These are discussed in section 6.

2.3.4 Unlinkability

Pfitzmann and Köhntopp (2000), in reasoning about anonymity systems, propose a viewpoint defined by a set of subjects sending messages to a set of recipients. In this setting, the critical concept is an *item of interest*, defined as the sending or receiving of a message.

The desirable property of an anonymity system is that items of interest are *unlinkable* to any identifier in the system, and no identifier in the system can be linked to a specific item of interest. This provides a basic definition of anonymity, however it does not lend itself to quantifying the property.

Bitwise unlinkability expresses the concept of unlinkability in terms of the cryptographic hiding of messages when they pass through a network node (Danezis, 2003). This hiding makes it impossible for an attacker to link an ingoing message to an outgoing message by examining the raw data itself. Almost all implemented anonymity systems use this process in combination with mixing (section 3) to achieve practical anonymity.

2.4 Pseudonymity

> "If liberty means anything at all, it means the right to tell people what they do not want to hear."
>
> – Eric Arthur Blair (writing as George Orwell)[2]

Closely related to anonymity is the concept of pseudonymity. Pseudonymity, which stems from the Greek 'πσευδος' (*pseudos*, false) again combined with 'ονιμα', refers to the adoption of a false name. This is also commonly known as an allonym ('αλλος', *allos*, other), *nom de plume* (pen name) or *nom de guerre* (name of war), after the traditional pre-computer use of pseudonyms as a method by which authors could publish politically inconvenient material without the threat of retaliation.

Pseudonymity, in terms of usable online systems, causes users to be associated with an identifier that is at least semi-persistent. The purpose of this approach is generally to allow types of transaction, relying on user history and behaviour, that are not possible using totally anonymous systems. This is of particular use in systems that rely on networks of trust between users, and thus cannot rely on a simple one-use "session identifier" approach.

[2] from "The Freedom of the Press", Orwell's original unpublished preface to *Animal Farm*, first published in *The Times Literary Supplement*, 15 September 1972.

As mentioned above, pseudonymity can be achieved through the use of an anonymous infrastructure with suitable user information and history stored within the explicitly transmitted data. If the communication system on which communication relies is inherently anonymous then pseudonymity becomes an easier proposition, as data can be released as chosen by the user without fear of extra information leakage from the system. Care must be taken that the interaction between deliberately released data and otherwise harmless data within the system cannot interact to reveal more than is intended.

Pseudonymity may therefore be seen as a problem that exists at a 'higher' level than anonymity. An anonymous channel may have some form of persistent user identification added that is kept secret between the sender and recipient. Pseudonymity may be viewed less as a primitive construction and more as a combination of other security properties such as secrecy, anonymity and authentication. There has been relatively little practical research into pseudonymity, perhaps due to the status of anonymity as a problem yet to be definitively solved.

Pseudonymity is related to the overall topic of censorship resistant systems. Interesting work into such systems may be found in the Eternity service (Anderson, 1996), the "Publius" system (Waldman et al., 2000), and the work of Goldberg (Goldberg, 2000; Goldberg and Wagner, 1998). These systems aim to provide an online cryptonymous data store that automatically mirrors documents across the entire network. Such approaches are designed to be resistant against any measure, whether physical or legal, to remove data from the system once it has been published.

2.5 Trust and Reputation

Trust and reputation are closely linked properties, particularly within the context of anonymity systems.

Reputation is a property that allows a user to make an informed decision about whether or not to trust another user. This ability is important in commerce systems where users are required to invest real economic interests in other users of a system. The potential risks of such a system are high, especially in cases where there are no legal restrictions on the parties involved in a transaction. In these cases, which are common where the Internet allows commerce between countries with differing legal systems, reputation may be critical to users and legitimate businesses. As such, in many of the practical applications of anonymity and pseudonymity, reputation is the key to a usable system.

2.5.1 eBay

The most famous reputation-based system in common use is the seller rating on eBay (eBay, 2004). eBay is a popular online auction site that manages the buying and selling of a large quantity of items all over the world. The site functions similarly to most auctions, in that buyers bid against each other during a given time period. At the end of this period, the item is sold to the highest bidder.

When the transaction is completed, both the buyer and the seller are encouraged to leave feedback concerning the behaviour of the other party in the transaction. This feedback takes the form of a positive, neutral or negative rating and an 80 character free-text comment for extra details on the transaction. When considering an item, buyers can first examine the ratings of the potential seller. This allows a user to make a decision on whether or not the seller is likely to be trustworthy and follow through with the transaction. The implicit assumption is that the higher the number of positive feedback reports a seller has listed, the more likely they are to be trustworthy.

A seller will endeavour to maintain and protect their reputation in order to attract more business in the future. As such, the seller will be unlikely to perform any action that could damage their reputation.

This approach towards trust management in commerce systems has been the subject of some study, for example in the work of Dellarocas (2001). Even before the invention of eBay and similar systems, reputation as a method of enforcing positive behaviour in markets had been well-known, and has received much attention in that field.

2.5.2 Numerical Reputation

Beyond the use of reputation as a human measure of trust, reputation may also be treated as a strict numerical property that allows judgements to be made based on objective measurements. A common use of this appears in peer-to-peer network systems (Dingledine et al., 2003), which must dynamically decide on the best routes for information flow. In such systems, reputation is a property based on the past reliability performance of the network node in question.

This use of reputation can be extended to relate to any property of interacting systems. Recent work in reputation, such as that by Engelmann and Fischbacher (2002), has sought to unify the traditional economic and psychological aspects of reputation with the newer reputation algorithms emerging from the field of computer science, such as those presented in (Dingledine et al., 2001; Dingledine and Syverson, 2002b; Dingledine et al., 2003).

In anonymity systems, which often rely on distributed networks of untrusted participants, reputation algorithms are of great interest in providing a degree of

assurance that network nodes will behave as advertised. Similarly, for pseudonymous online systems, reputation is of great importance in enforcing "good" behaviour between participants.

2.6 Repudiation

Security protocol designers often wish a party to prove a certain fact to another party in such a way that it cannot later be retracted or denied ("repudiated"). This allows an agent who negotiates a contract with another party assurance that this party will honour the contract. Non-repudiation is a commonly desired property in the field of security.

In anonymity systems, however, it may often be desirable to prove a fact to another agent at some point, but for this fact to be unusable in creating long-term profile information. The ability to present a piece of information to an agent, but for this information to be valid for no longer than the course of that single transaction, prevents personal information concerning the agent from being catalogued for the purposes of future identification.

An important approach towards this form of information exchange is the *zero knowledge proof* (Feige et al., 1987). This allows individuals to prove that they hold a certain piece of information without revealing the information itself. Whilst this notion is not intrinsically a feature of anonymity systems, it is a closely related information hiding property that is of great potential use in achieving anonymity in realistic systems. Zero knowledge proofs are discussed in section 4.4.

Chapter 3

Mixes

This section describes the most studied and implemented anonymity technology: the mix. The mix underpins almost the entire field of anonymity research. There are many variations that build on the basic mix, designed to overcome attacks against the structure and to improve on desired aspects related to performance.

Mix systems were first proposed by Chaum (1981) as an anonymizing process for electronic mail.

3.1 Basic properties

A mix node accepts incoming messages, often emails, which are encrypted with that node's public key. The mix decrypts the message and removes all sender information from the headers. This provides unlinkability, defeating recipient analysis by removing identifying information. When some specific condition is reached, the mix forwards a mixed batch of message on to the respective recipients, or to other mixes that repeat the strategy. Mixing defeats traffic analysis, by removing timing information. The condition that determines the sending of a message has the potential to greatly affect the strength of the anonymity provided by the mix and the attacks that can be performed against the users.

Latency is inherent in a mix system. This latency, often an hour or greater for a message travelling through a network of mix nodes, makes the system unusable by applications that rely on swift message passing. As such, mixes are typically used for message-based applications such as email or Usenet posting.

3.1.1 Unlinkability

Mixes accept cryptographically secure messages, which are decrypted and forwarded to the receiver. This prevents an attacker from observing any similarity

between the ingoing message and outgoing message.

The message sequence from a sender to a recipient through a mix can be expressed as (where $\{M\}_K$ refers to the message M encrypted with the key K)

$$\begin{array}{rcll} \text{Sender} & \longrightarrow & \text{Mix:} & \{\text{Msg, Recipient}\}_{K_{Mix}} \\ \text{Mix} & \longrightarrow & \text{Recipient:} & \text{Msg} \end{array}$$

This behaviour can be extended to many mixes by including multiple layers of encryption, each of which corresponds to a mix in a chain. Messages can therefore take the form:

$$\{\{ \text{ ... } \{\text{Msg, Recipient}\}_{K_{Mix_n}} \text{ ...}\}_{K_{Mix_2}}\}_{K_{Mix_1}}$$

3.1.2 Mixing

If messages left the mix a known time after they entered, it would be possible to deduce which messages corresponded to which recipients. So a mix eliminates correspondence between the time of the message entering the node and the time at which it leaves. This is achieved in a number of ways, most usually by storing a number of messages and then flushing them in a batch, with internal reordering (mixing) of the messages taking place.

The trigger conditions that cause mixes to send out their batches of email are the subject of much study. Different approaches appear to be vulnerable to different attacks, and the problem is by no means trivial. In the following section we review a number of the common mix designs with reference to their flaws and advantages. A useful overview of this subject is given by Diaz and Preneel (2004b).

3.2 Pool Mixes

The original mix proposed by Chaum (1981) achieves mixing by accepting messages that are stored until a certain threshold is reached. The mix then sends out all stored messages, with the messages reordered from their arrival order.

The technique of storing a number of messages until a given condition is reached and then flushing messages is the strategy behind a family of mixes known as pool, or batch, mixes. (Another approach is the “continuous” or “stop-and-go” mix proposed by Kesdogan et al. (1998), discussed in section 3.3.)

There are two major conditions that trigger the flushing of a set of messages:

3.2.1 Threshold Mixes

The messages entering the mix are placed into a pool, and when the number of messages in that pool reaches a certain threshold, the mix flushes the messages.

A mix using this strategy may, however, retain a certain number of messages at each round. Every message entering the mix therefore has a small probability of staying in the mix for an arbitrarily long time. This effectively increases the size of the anonymity set for that mix node to infinity, as the set of messages that could contain the desired message is all messages that leave the mix after the desired message entered it. This increases the latency for message delivery.

Threshold mixes are open to an "$n-1$ attack", in which the attacker floods the pool with a set of fake messages that are traceable to the attacker. This allows the attacker to input all but one message for each round, and thus trace the remaining message. This attack is discussed in section 3.5.

3.2.2 Timed Mixes

The messages in the pool are flushed at a given time interval (Serjantov and Newman, 2003). This prevents the possibility of a message remaining in the mix indefinitely, and overcomes problems that arise if too few messages are processed by the mix to reach a number related threshold. This guarantees a lower bound on the latency. However, if only a small number of messages, or even an individual message, have entered the mix in a given time period then the anonymity set for the messages is severely compromised.

Most implemented systems that use a pool mix combine a timing and a threshold constraint in order to reap the benefits of both strategies. A notable example is the MixMaster anonymous remailer network (Möller et al., 2003), discussed in section 5.3.

3.3 Continuous Mixes

Kesdogan et al. (1998) present the concept of a *stop-and-go*, or *continuous*, mix. A sender selects a delay from an exponential distribution and adds it to the message sent to the mix. The mix delays the message for the given time period before forwarding it.

This approach allows a great amount of flexibility in the latency of messages. The user can balance the strength of the provided anonymity against the latency. This gives the continuous mix potential use in applications that require tighter time constraints than is possible to achieve with other forms of mix. In addition, a user

has a predictable latency for messages and so can potentially detect interference in the network from an attacker who is delaying messages.

3.4 Mix Networks and Mix Cascades

It is not desirable to rely on a single mix node for anonymity. The possibility of traffic analysis is much higher for a single location on the network (Pfitzmann and Waidner, 1985). In addition, a lone mix node is a single point of failure. The possibility of the mix being compromised, by an attacker or a dishonest mix operator, is too high to risk the total loss of anonymity that these situations would cause.

So messages are usually routed through several mixes. This ensures the anonymity of a message, provided that at least one mix on the route is uncompromised.

The routing strategy for a message through such networks is important. *Mix networks* (Rennhard and Plattner, 2003) use a free-route strategy, and *mix cascades* (Dingledine and Syverson, 2002b) restrict the path a message may take through the network. There is debate as to which strategy is more effective and robust; Berthold et al. (2000) argue the case for using a mix-cascade over the more common mix-network. The relative advantages and disadvantages of both approaches are discussed here.

3.4.1 Mix Networks

Mix networks use a free-route approach to the network path. The user chooses a path from the available mixes for each message sent.

This provides flexibility. A user may choose to route their message through mix networks that they trust, or that have behaved reliably in the past. Free-route networks scale well as the number of network nodes increase. The free-route approach also provides fault-tolerance, as nodes that cease to function can be routed around by the system.

The free-route approach does, however, suffer from a number of potential active and passive attacks that can compromise anonymity. These are described in detail by Berthold et al. (2000). These attacks are possible for an attacker who controls the majority of mixes in a network.

It is possible for an attacker to partition messages going through an honest node based on the number of other nodes through which they have already passed, this information being available to a global attacker observing network traffic through compromised nodes. With successive rounds, this partitioning can reduce the anonymity set for a message to only those messages that have passed along the same route.

Danezis (2003) notes that in a free-route mix network with a large percentage of compromised nodes, the chances of a choosing an entirely compromised path become high. In such a compromised path, anonymity is completely lost.

Other attacks on free-route networks include the ability to link senders and receivers across an honest node, and the *trickling* and *blending* attacks (discussed in section 3.5).

3.4.2 Mix Cascades

Mix cascades (Pfitzmann and Waidner, 1985) use a number of fixed paths through the network. This ensures that more messages take a particular path than with a mix network, which prevents the partitioning attack.

In a mix network, a node may be a member of a large set of potential routes, so there may be a big difference between the maximum and minimum observed traffic. At some times a node may be overwhelmed by the traffic passing through it, while at other times it may be handling only a tiny amount of traffic. In a mix cascade, however, all the traffic is routed consistently through nodes in the path. This helps to ensure that traffic is as uniform as possible, which helps to prevent an attacker from taking advantage of patterns in message flow.

Mix cascades suffer from a number of problems. In a free-route network, the attacker must compromise a large number of nodes in order to successfully mount many of the attacks discussed; in a mix cascade only the nodes on a particular route need be compromised. This allows an attacker to focus resources. Also, the flexibility and fault-tolerance seen in free-route networks is to some extent lost in cascades. Finally, cascades are subject to a simple denial of service attack in which one node in a route is shut down, effectively destroying a significant portion of the network.

3.4.3 Hybrid Approaches

Neither of the two extremes offers the ideal solution to routing within a network of mixes. A combination of the two approaches may be a better solution. For example, Berthold et al. (2000) enumerate a number of proposed advantages to free-route networks and show that each can, to some extent, be achieved within a cascade.

Many of the techniques to improve the behaviour of cascade networks move away from the strict single-route towards a more flexible routing system. Danezis (2003) proposes to provide a number of cascade routes through the system, with the user selecting one of these. Another approach (Berthold et al., 2000) is to allow each node in a path to nominate a number of potential next steps. The user can then select from these options.

One hybrid approach uses a mix network based on a family of sparse, constant degree graphs known as "expander graphs" (see Danezis (2003) for details). This network topology, and the use of probabilistic techniques for selecting the next node in the message path, provide a system that seems to hold many desirable characteristics of both free-route mix nets and mix cascades.

3.4.4 Synchronous and Asynchronous Batching

Dingledine et al. (2004b) suggest that the perceived difference between mix networks and cascades is due to the synchronicity of batching rather than to the network routing. They present an analysis of three network topologies (free-route, cascade, and a hybrid "stratified" topology) that process messages synchronously. Every mix forwards messages at the end of a given *batch period*, and queues any messages that enter the network during such a period. The details of such a network, where synchronous batching is the defining characteristic of the network, were first given by Dingledine et al. (2001).

Dingledine et al. (2004b) show that the use of a synchronous batching strategy in free-route networks provides stronger anonymity overall than the other approaches suggested, and that the traditional problems with free-routes may also be to some extent avoided. This includes attacks that could previously be resisted only by mix cascades. The paper does, however, stress that comparison of the relative anonymity in these networks is open to further research and interpretation.

3.5 Attacks on mix systems

We now review the attacks that are possible against the anonymity provided by a mix network.

Attacks on anonymity systems seek to weaken the anonymity of participants in the network, or to render the network unusable so that anonymous messages cannot pass. The majority of these attacks rely on *traffic analysis*: the observation of the flow of messages in an attempt to learn sender or recipient identities (Back et al., 2001). These attacks may arise purely from passive "packet-sniffing" approaches, or an attacker may inject their own traffic in an attempt to reveal underlying patterns in the flow of data. Here we provide a high level review of the more well-known attacks against mix-based systems. A comprehensive survey of attacks is given in Raymond (2000).

3.5.1 Passive Attacks

There are two major threat models considered for mix networks. The first, and weaker, of these is the global passive attacker, who has the ability to observe all packets on the network, but not to inject, capture, delay or otherwise interfere with traffic. All attacks possible for this attacker, therefore, rely on observation of the information about senders and recipients that leaks from the unaltered behaviour of the network.

The *intersection attack* (Berthold and Langos, 2002) relies on the fact that user traffic is not random and can often have predictable behaviour within certain time periods. For example, consider pairs of users sending emails rapidly back and forth as a conversation takes place. This behaviour may take place only during certain time periods (during working hours for a business conversation, perhaps). By observing this behaviour and performing successive set intersections on the active users at particular times, it is possible to compromise user anonymity.

Other attacks available to the global passive attacker follow a similar form, referred to by Raymond (2000) as "contextual attacks". Any feature that distinguishes a particular message from the noise of the background traffic can be used to infer information about the sender or receiver. Uniformity is the key to defeating the global passive attacker.

3.5.2 Active Attacks

Far more dangerous than the global passive attacker is the local[1] or global *active* attacker. This attacker can inject their own messages and packets into the data stream, to compromise a number of mixes (up to total control of the network, whereupon the attacker becomes unconditionally able to trace a message), to delay messages passing through the network, and to alter messages that are already passing through the network.

A comprehensive overview of active attacks on mixes is given in Serjantov et al. (2002). Here we review the more common approaches.

The majority of attacks that an active attacker may attempt against a mix are known as *blending attacks*. Blending attacks, including the $n - 1$ attack, allow a sufficiently powerful attacker to identify the receiver of a message with absolute certainty for basic pool mixes, and probabilistically for some more advanced mix designs. Blending attacks take the form of *trickle* attacks or *flood* attacks (Serjantov et al., 2002).

Trickle attacks rely on the ability of an attacker to delay or delete messages approaching a mix. The attacker waits until a mix has flushed its contents then

[1]The "local" attacker refers to an attacker who does not have total coverage of the network, however is not necessarily limited to a single machine.

prevents all, or almost all, messages except the target from entering the mix in the next round. This attack is less effective with more complex mixes that use dummy traffic and advanced pool-flushing algorithms.

Flood, or $n-1$, attacks also wait until the pool of a mix is emptied. After allowing a single target message through, the attacker then floods the mix with enough messages to cause the pool to be flushed. As the attacker can recognize their own messages, the target message is easily traced.

The $n-1$ attack is particularly resistant to attempts to reduce its effect. The use of complex pool flushing algorithms that retain a number of messages at each round and dynamically alter timing conditions for flushing only degrade this attack to tracing messages with a certain probability of incorrect identification. However, the attack relies on a very powerful attacker model, as the $n-1$ attack must be performed on all mixes within a network to be fully effective. Thus, an attacker must be able to flood an entire network with dummy messages whilst observing the passage of the desired message through the network.

3.5.3 Denial of Service Attacks

Another category of attacks against mix systems or networks does not seek to reduce or compromise the anonymity of users, but rather to prevent the functioning of the mix networks. These denial of service attacks are of great concern in mix cascades that use known chains of mixes. Destroying the functionality of a single mix node removes an entire route from the cascade. A determined attacker could have a serious effect on a network by attacking a small percentage of nodes in the network.

The precise methods are much the same as for any other denial of service attack. The focus on messages in mixes lends itself to flooding nodes to the extent where valid messages cannot pass, however simple computer hacking or distributed denial of service attacks from worms would achieve much the same goal.

3.6 Dummy Traffic

In any proposed mix design there may be fewer messages passing through a system than required for anonymity, which relies on having as many interactions occurring in the system as is possible. Low traffic makes most attacks on the anonymity of users far simpler. *Dummy traffic* (Berthold and Langos, 2002) can be inserted into the network to confound traffic analysis.

Dummy traffic is usually generated by the mix systems themselves, although it is possible for end users to generate their own dummy messages. The more

dummy messages passing through a network, the better for anonymity. Some systems pad traffic with dummy messages to cause constant message flow at all times in the system (Dai, 1996).

Flooding the network to capacity with dummy traffic overlaying genuine messages is not feasible, however. Many implemented systems, such as (Möller et al., 2003), use a lower level of dummy traffic, generated by the mix nodes themselves. This traffic is created according to a probability distribution that aims to mimic real traffic, although the best probability distribution to achieve this is not known.

There are a number of design decisions when creating dummy traffic. Assuming that the mix itself generates the traffic, it must be decided whether to insert the dummy messages into the message pool, thus potentially affecting the trigger condition for flushing the mix, or to add messages to the traffic as it leaves the mix. There are also choices as to whether the dummy traffic should adapt as the message flow increases, or remain constant. These issues are discussed in detail by Diaz and Preneel (2004a), Diaz and Preneel (2004b), and Möller et al. (2003). In all cases, the desired behaviour of dummy messages is to be indistinguishable from the normal traffic in the network. Describing this normal traffic is therefore a highly important task in the design of mix networks.

3.6.1 RGB Dummy Traffic

A scheme proposed by Danezis and Sassaman (2003) routes dummy traffic through the network, and delivers it back to the originating node. This allows the original node to detect attacks on the network by measuring the traffic that is successfully returned.

The name *RGB* refers to the types of traffic that are used to detect flooding attacks. *Red* traffic is "heartbeat" messages that are regularly sent through the network with a final destination of the sending mix. *Black* traffic is the traffic received by the mix, which may be either valid messages (including the dummy traffic of another mix) or an attacker's flooding messages. *Green* traffic is dummy traffic inserted by the mix in response to estimates concerning the level of flooding that the node is experiencing.

This uses the fact that many active attacks, such as the $n-1$ attack, use denial of service to achieve their aims: to control the messages entering or leaving a mix, it is usually necessary to flood one or more other nodes in the network. In a denial of service attack, the profile of the dummy traffic changes as packets are dropped. Although dummy traffic is designed to appear uniform to an outside observer, the author of the dummy packets can always distinguish them. The RGB approach takes advantage of this property to provide a warning mechanism for users. In the case that fewer "heartbeat" messages are returned than expected, a user may

assume that an $n-1$ attack is being mounted, and either cease operations or inject an amount of "green" dummy traffic to improve the anonymity of valid messages. While this behaviour results in a denial of service for the user, it is preferable to the loss of anonymity otherwise caused by the $n-1$ attack.

Chapter 4

Other Technologies

Despite the focus on mixes within the anonymity community, there are other approaches that seek to achieve anonymous communication. Many of these approaches suffer from flaws either in their scalability or in the level of anonymity that they provide. For this reason most are not the subject of intensive ongoing research, although most have utility in specific application domains.

4.1 Dining Cryptographer Networks

In 1988, seven years after the mix (Chaum, 1981), Chaum proposed another scheme for anonymous message sending. The "dining cryptographer" network provides unconditional sender and recipient untraceability within a network, given assumptions of cryptographic security between parties Chaum (1988).

Consider three cryptographers dining in a restaurant. At the end of the meal, they are told that an anonymous party has paid for the evening. They wish to know whether one of them or an outsider is their mysterious benefactor. They each flip a coin that is seen by both themselves and their right-hand neighbour. They each state whether the results of the two coin flips that they have seen are the same or different. If they are the diner who paid, they lie about the result. Then, if there is an even number of 'same' replies, one of the diners is lying. So the desired information can be deduced, but without revealing the identity of the buyer.

It is possible to generalize this approach to a reliable broadcast network where participants share cryptographic keys. The results of the hypothetical "coin flip" are broadcast one bit at a time into the network. If a participant wishes to make a message public, they flip the status of their broadcast bit for those bits that correspond to the high bits of their message. Thus, for each round a "default" parity of the broadcasted bits corresponds to a "0" bit in some message, and an inverse parity of all broadcasted bits corresponds to a "1" bit in the message.

Chaum (1988) proves that, under the assumption of a reliable broadcast channel and collusion of no more than $n - 2$ participants in the network, this approach provides unconditionally secure sender and recipient anonymity.

Dining cryptographer networks are impractical for large-scale systems. The requirement of a reliable broadcast channel between participants causes the protocol to be fragile against active attackers who can flip bits and thus potentially gain information. Some work has been carried out to resolve this problem (Waidner and Pfitzmann, 1990), however there is no entirely satisfactory solution.

In addition to being fragile, the dining cryptographers network is inefficient. A message is passed at the rate of only one bit per round on the network. In addition, the protocol relies on a large network of shared keys. Approaches to solving this problem are presented by Chaum (1988), however the logistics of achieving this on a large-scale network are far too complex for any widespread implementation to be considered.

Despite the inefficiencies of dining cryptographer networks, they do provide an effective method to unlink actions and actors. For small-scale systems, such as board room voting, an implementation of the dining cryptographer network could find use.

4.2 Onion Routing

Onion routing aims to overcome the high latency inherent in mix systems. The design was proposed by Goldschlag et al. (1996) as a method for hiding routing information in applications that demand almost real-time network connections.

The approach is to wrap a data payload with successive layers of encryption, corresponding to nodes in a chain of communicating servers, to form an "onion". This encryption is performed by the first node of a routing network, which thus acts as a proxy server for future traffic. Each node along the route "peels off" a layer of encryption, and forwards the resulting onion the next node in the chain, as revealed by the decryption process. To defeat analyses of the decreasing size of the onion, each node pads the outgoing onion up to a constant size.

This differs from a mix system (section 3) in that an onion routing node does not perform any mixing operations within the node; each onion is immediately forwarded to the next node.

The network traversal of an onion causes a "virtual circuit" to be set up along the route taken. Each node stores the forward and reverse nodes in the chain along with appropriate decryption keys for forthcoming data. The initiator of the connection encrypts its data with the successive encryption keys of the routers in the chain and sends the data. Each onion routing node decrypts its own layer and forwards the data to the next node in the chain. As each node only knows the next

router in the chain, it is impossible for an attacker to determine the entire path without compromising either the entire chain or the end-point nodes.

Onion routing is the subject of ongoing research (Syverson et al., 2000; Clayton, 2003; Dingledine et al., 2004a) as a method to achieve relatively strong anonymity without the latency overheads of mixes. For details on a current implementation of onion routing, see section 5.6.

4.3 Crowds

The crowds system (Reiter and Rubin, 1998) is proposed as a method for providing anonymous web-browsing capabilities. The system relies on a network of nodes, referred to as *jondos* (*John Does*), that forward web requests for other participants.

The crowds system re-routes encrypted web requests randomly between users of the network to obscure the location of the original request. When a user attempts to access a website through the network, the request is forwarded to a jondo. With a certain probability, each jondo forwards the request to another jondo, or fetches the website itself and returns the information to the original requester.

The crowds system is vulnerable both to the global passive attacker and to corrupted jondos. A global passive attacker can observe the flow of a message request and trace the originator, while corrupted jondos can collude or work individually to increase the likelihood that a given member of the crowd originated a request (Shmatikov, forthcoming).

Due to the poor anonymity guarantees of the crowds system, which are an inescapable aspect of the design, crowds has not gained much favour within the anonymity community. However, it does provide stronger anonymity than ordinary web-browsing and would appear to be scalable, and the weaknesses against determined attackers have not prevented an implementation of this approach from becoming more generally popular (see section 5.7).

4.4 Zero Knowledge Proofs

Zero knowledge proofs are an information hiding property with potential use in many applications that rely on anonymity.

Zero knowledge proofs allow an agent to demonstrate with arbitrarily high probability that an assertion concerning the agent is valid, without revealing any more information than the validity of the assertion (Feige et al., 1987). The proofs typically take the form of a “commitment” stage from the participant who wishes

to prove their knowledge, followed by a challenge-response initiated by the other party.

The capability of these proofs to allow assurance without revealing information has uses in digital anonymous cash, in authentication without revelation of identity, and in verification of adherence to protocols by possibly untrustworthy participants.

The canonical example of a zero knowledge proof is that of Ali Baba's cave (Quisquater et al., 1990). Consider a cave containing a hidden door that allows a person knowing the secret magic words to pass from C to D (see figure 4.1).

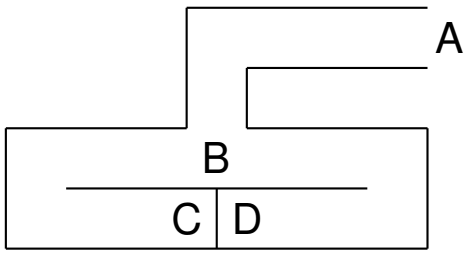

Figure 4.1: Zero knowledge proof: Ali Baba's cave with a secret door between C and D

If Alice wishes to convince Bob that she knows the secret to pass between C and D, she can demonstrate her knowledge, without revealing the secret of the door to Bob, in the following way.

Initially, Alice and Bob stand together at the entrance of the cave, A. Alice proceeds to either C or D without being observed by Bob. This is the *commitment*.

On reaching her destination, Alice calls out to Bob, who then proceeds to stand at B. Bob then calls out a direction from which he wishes Alice to appear. If Alice knows the secret of the door, she can appear from either direction regardless of her initial commitment.

At each run of this protocol, Alice has a fifty percent chance of being in the correct location. If Alice ever appears from the wrong direction, Bob can be confident that she does not know the secret of the door. Each time that Alice appears from the correct direction, Bob's confidence in Alice's knowledge of the secret increases. By performing this test a large number of times, Alice can convince Bob of her knowledge of the secret to an arbitrarily high level of probability without ever revealing the secret itself.

Only Bob, or an observer present at the time of the protocol run, can be confident in the results. Any replaying of the protocol after the event, such as a video recording of Alice emerging from the correct directions, could be faked.

Chapter 5

Deployed Anonymity Systems

Chaum (1981) sparked serious research into online anonymity. Since then, several systems have been implemented. The most famous historical anonymity services, and the leading contemporary approaches, are reviewed in this section.

5.1 Type-0 (`anon.penet.fi`)

The original, and to this day most popular, anonymous remailer system was a single machine, `anon.penet.fi`, run by Johan Helsingius in Finland. This machine operated a simple remailing policy whereby a user could send the server an email containing an extra header specifying the ultimate destination of the message. The server stripped all identifying information from the email's headers, then forwarded the email to the destination. A simple reply service was also provided, by which a user could set up a pseudonym on the server, all email to which would be forwarded to the owner's real email address.

This system, though naïve, was tremendously popular. However, the system ran on a single server, which was open to both legal and electronic attacks. The nature of the server also caused the administrator to be in possession of the identities of all users. This simplicity of use and lack of protection against attacks that are now considered commonplace has caused `anon.penet.fi` to be retroactively labelled as the "Type-0" remailer.

A user of the service began anonymously distributing material from among the secret teachings of the Church of Scientology, which responded by suing the user. The legal case caused the logs and data stored on `anon.penet.fi` to be subpoenaed. This, in combination with allegations of the distribution of child pornography through the server, caused its eventual closure in 1996, after three years of providing anonymous services.

5.2 Type-I (Cypherpunks)

Members of the online cryptography advocacy mailing list `cypherpunks` (Cypherpunks, 2004) developed and deployed the "Type-I" or "Cypherpunks" remailer network. This network uses multiple servers and PGP encryption for all messages. By encrypting emails with the keys of a number of servers it is possible to chain a message through a series of nodes before reaching the final destination. These design decisions removed the critical flaws that eventually caused the demise of the original Type-0 remailer.

The Cypherpunks remailer system, still in use today (APA-S, 2004), uses *reply blocks* to allow anonymous replies to emails. A reply block is a series of routing instructions that allow the message to be delivered to a pseudonym. This data is included in an anonymous message when sent, and so allows the recipient to reply, despite not knowing the identity of the sender.

5.3 Type-II (MixMaster)

The Type-I network is still open to many attacks on the anonymity of users. In 1994, in an attempt to rectify these problems, Lance Cottrell released the Type-II "MixMaster" remailer (MixMaster, 2004; Möller et al., 2003).

Mixmaster improves the anonymity service over the original Type-I remailer. Firstly it introduces fixed-length messages, with shorter messages being padded up to the message length and larger messages being broken up into multiple messages. This defeats the trivial attack of following a message from sender to receiver through the size of the message. Secondly, it implements a message pool to increase the difficulty of performing traffic analysis. This makes MixMaster the first implemented system to approach the design laid out by Chaum (1981).

MixMaster development has continued since 1994, and it is the most popular remailer network currently in use. MixMaster does not implement reply functionality, due to the difficulty of maintaining anonymity with the use of reply blocks, so the Type-I remailer also continues to be used.

5.4 Type-III (MixMinion)

MixMinion is the latest remailer network, jointly designed by members of the academic and remailer communities. MixMinion improves on the Type-II remailer by including a secure reply capability (Danezis et al., 2003).

Many of the features used in Type-II remailers are employed to transport mes-

sages across the network, with the addition of functionality such as TLS[1] encryption for links between mixes and a frequent rotation of authentication keys for each mix in order to prevent replay attacks.

MixMinion includes secure reply for anonymous messages. This is achieved through *Single-Use Reply Blocks* (SURBS). The message path is split into two halves (or "legs"), and the first half of the message path is encoded into the SURB, which accompanies the message. The message then continues along the entire path. When an anonymous reply is formed, the path encoded in the SURB is used to determine the second half of the reply path (which thus returns the message to the original sender), and the first half is formed randomly by the mix. Hence forward and reply messages are indistinguishable within the network.

MixMinion is still in the early stages of development, however the project is receiving considerable interest within the remailer community and development is highly active.

5.5 JAP

JAP (2004) is an anonymizing proxy server developed and hosted at the Technical University of Dresden, with certain nodes on the network hosted by other Universities. The JAP software uses a mix cascade approach to provide anonymity, however the application of the network is geared towards low-latency requirements such as web browsing.

JAP gained some notoriety in 2003 when the German Federal Bureau of Criminal Investigation successfully ordered the JAP service to record data concerning connections for future analysis. This ruling was later overturned by the District Court in Frankfurt (JAP Ruling, 2004), however the damage done to the system's reputation and the worries concerning future reversals of the court's decision have remained.

5.6 Tor

Tor ("**T**he **O**nion **R**outer") (Dingledine et al., 2004a) is an implementation of an onion routing network designed to allow anonymous access to lower-latency services such as web-browsing and remote shell access. This network, along with the MixMinion effort, is being developed as part of the FreeHaven project to create a functional data haven (Dingledine et al., 2000).

[1]The Transport Layer Security (TLS) protocol is the latest incarnation of Netscape's Secure Sockets Layer (SSL) protocol. TLS is an extension of version 3 of the SSL protocol, and has been adopted as a web standard. TLS is commonly used in most forms of encrypted Internet traffic.

Tor presents a scalable implementation of onion routing that uses standard proxy protocols to forward services across the network. The system is in the early stages of development, but is already usable for a number of applications. The combination of the low-latency overheads of the network with the integration of the system into a standard proxy format may allow Tor to be more widely adopted than many anonymity services, which have traditionally been weighed down with complex requirements on the user.

Tor can provide not only anonymity for the user of the system, but also can obscure the location of services such as web sites offered through the network. This is achieved through the ability of a server to "extend" a circuit into the network with no fixed client on the other end (see section 4.2 for details of circuits in onion routing). A client using Tor can connect to this extended service and relay requests through to the server without knowing the server's address. This provides censorship-resistance capabilities.

5.7 Freenet

Freenet (Clarke et al., 2000) is an anonymous document publication and distribution service that uses encrypted data storage, geographical distribution, and anonymous communication between nodes. Anonymity within the network is provided both to the authors and the readers of documents.

The Freenet network protocol is most similar to the Crowds system (Reiter and Rubin, 1998), with requests being randomly redirected between nodes before finally reaching their destination. As such, the anonymity that the system provides is vulnerable to most of the attacks that mixes and onion routing seek to prevent.

Despite these flaws, Freenet has received a great deal of interest and use. The project homepage (FreeNet, 2004) claims over two million downloads of the Freenet software as of the time of writing, and there is a significant user community hosting content within the Freenet system. The popularity of the system represents more interest than in any other anonymity-based service currently running on the Internet.

Chapter 6

Modelling Anonymity

It is common for analytical approaches to describing anonymity systems to use simple quantifications and basic probabilistic models. There have also been some approaches to producing formal frameworks for the more general description of anonymity systems. These approaches, while not in widespread use, provide some inspiration for methods that could be produced in the future. Here we review some notable approaches.

6.1 Quantifications

Anonymity is not all or nothing: some anonymity systems are in some sense stronger than others. Here we review some means to *quantify* that strength.

6.1.1 The Anonymity Set

The anonymity set has been the basis of the traditional measure of anonymity since it was proposed by Chaum (1981). The metric is the cardinality of the set S of all participants who could have initiated an action (usually, the sending or receiving of a message). Assuming a uniform probability distribution, the probability that any individual i is the initiator is $P(i) = 1/|S|$.

This provides a useful first approach to measuring anonymity, but it is not sufficiently powerful to describe the anonymity of a real system. For a set of users with non-uniform probabilities of having performed an action, the anonymity set is a poor metric of anonymity.

> "Anonymity is the stronger, the larger the respective anonymity set is and the more evenly distributed the sending or receiving, respectively, of the subjects within that set is." Pfitzmann and Köhntopp (2000)

6.1.2 Information Theoretic Metrics

In an attempt to overcome the limitations of the anonymity set as a metric, Serjantov and Danezis (2002) and Diaz et al. (2002) independently proposed information theoretic anonymity metrics.

Information entropy (Shannon, 1948) quantifies the level of uncertainty inherent in a set of data. Shannon entropy, in bits, may be calculated as:

$$H(S) = -\sum_{i \in S} p_i \log_2 \ p_i \tag{6.1}$$

where p_i is the probability of event i from the set of all possible events S.

The information theoretic anonymity metric calculates the entropy in terms of the *anonymity probability distribution* over the set of all users. This probability distribution links each user with a given message through a (possibly zero) probability of being either the sender or the receiver of that message, as viewed *a posteriori* by an attacker.

The information theoretic metric expresses anonymity in terms of the number of additional bits of information an attacker would need to perfectly identify the originator of an action. If a particular $p_i = 1$, hence all other $p_j = 0$, then $H(S) = 0$, and an attacker needs no further information to uniquely identify i as the source of the action in question. In the case that all the p_i are equally probable, $H(S) = \log_2|S|$, and the metric reduces to the simple anonymity set metric.

This approach provides a flexible and robust approach to quantifying anonymity properties. There is further work to be undertaken concerning the analysis of the anonymity of chained mix nodes (Serjantov and Danezis, 2002). This metric suggests the possibility of measuring attack efficacy via the entropy reduction caused by the actions of the attacker, however this capability has not been explored in the literature.

6.2 CSP and Anonymity

Schneider and Sidiropoulos (1996) examine anonymity as formalized in the CSP notation (see section 7.3.1).

The model relies on using certain existing features of CSP as an analogy for aspects of anonymity. "Hiding" CSP events from the view of other processes is used to model the anonymous sending of a message. Parallel execution of processes models an anonymity set of processes that could have performed an action.

The anonymity property is defined to be that the *traces* of the model (the sequences of actions observable to an outsider or attacker) should be identical, re-

gardless of the original sender of the message. This ensures that an attacker cannot distinguish a user in the network based on the actions that the users perform.

This CSP anonymity framework is then used to prove the unlinkability between sender and message in Chaum's basic three agent dining cryptographer network (Chaum, 1988). The model is highly specialised for the specific case of a dining cryptographer network, and has only the broadest applicability to other anonymity systems.

Adding a probabilistic aspect to the more deterministic views that are traditional in process calculi appears to be an essential aspect of building a functional model of real anonymity-providing services. Schneider and Sidiropoulos (1996) propose this approach, but it does not appear to have been subsequently pursued.

6.3 Group Principals

Syverson and Stubblebine (1999) present an approach towards formally reasoning about anonymity systems based on epistemic logic (see section 7.2 for a discussion of such logics).

Their approach treats the knowledge and actions of a *group* of principals, rather than that of individual agents. This shift from individuals to groups is well-suited to modelling anonymity systems, which intrinsically rely on the interaction of groups of agents in order to preserve anonymity. The epistemic logic developed in the paper relies on the standard $S5$ axioms for modal logics (see section 7.2), and also gives an agent a local "clock" that keeps track of the time-order of events observed by the agent.

The model allows principals to possess knowledge from three sources:

- A *log* of timestamped actions, from which elements may be deleted
- A set of *facts*, either deduced from the environment, or predefined
- A set of recent *actions* performed by the agent

The logic expressed by Syverson and Stubblebine (1999) relies on four *group principals* that allow the expression of knowledge possessed by a group of agents. These principals are:

- Collective Group ($\star G$): What is known by combining the knowledge of the group G of agents.
- And-Group ($\&G$): What is known by every member of group G.
- Or-Group ($\oplus G$): What is know by one or more members of group G.

- Threshold-Group ($n - G$): What is collectively known by any given subgroup of G of cardinality n; this may be considered as a collective group of or-groups each with cardinality n.

This framework treats actions that are performed within a run of the system, and are entered into or purged from the log of any agent that observes the action. A set of axioms based on the group principals listed above allows agents to gain knowledge from the system as each action is performed. The use of deduction rules expresses the knowledge that a particular agent may gain, and thus the potential of an attacker to deduce the identity (compromise the anonymity) of an agent in the system.

6.4 Anonymity in Multiagent Systems

Halpern and O'Neill (2002) present a "runs and systems" framework for analysing security systems. This is based on an epistemic logic of multiagent systems. Each agent in a system consists of a local state that contains all knowledge held by that agent. The state of the system is a tuple of the local states of all agents in the system.

The basis of the modelling approach is that of the *run*, a description of the possible execution states of a system considered over time. Logical deductions concerning the properties of agents are made based on *points*, expressed as a pair (r, m) of a run r at a certain time m. This is analogous to the concept of *traces* seen in the process calculus CSP (see section 7.3.1).

Halpern and O'Neill (2004) then apply this framework to the problem of analysing anonymity properties within multiagent systems. The definitions of anonymity represents the lack of knowledge that an agent j has to link an agent i with a particular action a. If j cannot deduce a link between i and a then i can be said to have anonymity with respect to a.

This definition is extended to two forms: *minimal* and *total* anonymity. With minimal anonymity for a particular action, j is unaware that i was the originator of this action, but there are no limitations on how many other agents j believes could have performed the action. For total anonymity, then for all $i \neq j$, j believes that i could have been the source of the action.

This approach includes a probabilistic element to the anonymity property. This is achieved by defining a probability associated with the knowledge of an agent at each stage. Thus, runs of the system for each probability are considered with varying degrees of likelihood. This attempt to address the probabilistic aspects of an anonymity system are some of the first that have been seen in formal methods approaches to anonymity.

6.5 A Modular Approach

Hughes and Shmatikov (2004) model anonymity based on the concept of a *function view*. This uses partial information known about some function as a model for aspects of anonymity. An attacker's knowledge is modelled through the information they hold concerning particular functions. This approach does not consider any probabilistic elements.

Hughes and Shmatikov (2004) model and quantify anonymity using epistemic logics and process calculi combined in a modular fashion. (For more on these, see section 7). Epistemic logic is used to model the system itself, representing all possible states of a system as a Kripke structure[1]; observational equivalences from process calculi are used to express the observable differences between configurations of the system.

The representation of anonymity revolves around the notion of "opaqueness", defined as the information that an attacker or observer may gain concerning a particular function within the framework of the function view. Increasing levels of opaqueness hide greater amounts of information in the function, which correspond to greater levels of uncertainty about which aspects of a system are linked.

Hughes and Shmatikov (2004) present a case study using this framework. The study considers an anonymity property, defined as the secrecy of the identity of communicating agents, and a privacy property, defined as the secrecy of the relationships between agents. The proof that these properties hold is highly involved.

This modular framework approach is flexible and presents some interesting approaches towards the definition of anonymity. The use of both epistemic logic and process calculi as modules in an overall approach provides insight into the potential rôles that they may play in developing other frameworks for modelling and measuring anonymity.

[1] A Kripke structure is a commonly used construct in logic. The structure itself is a non-deterministic finite state machine with all states in the machine possessing boolean labels that express the evaluation of that state.

Chapter 7

Formal Methods

> "Beware of bugs in the above code; I have only proved it correct, not tried it."
>
> — Donald Knuth. From a memo: *"Notes on the van Emde Boas construction of priority deques: An instructive use of recursion"*, (1977)

Formal methods provide a rigorous approach to defining required concepts, and thus the design and evaluation of software. By using mathematical notation to describe systems, formal methods aim to increase reliability and verifiability in software from the requirements phase onwards. This section reviews the use of formal methods approaches in security, with a focus on possible applications for the design or description of anonymity systems.

The field of formal methods encompasses many approaches for designing different kinds of system. Many of the most well-known formal methods are aimed at the specification and design of programs, and do not fall within the scope of this review, which chooses to concentrate on formal methods applied to security in communication protocols. As such, the reader familiar with formal methods should not be surprised at the lack of mention afforded to approaches such as B, Z or VDM below.

7.1 Motivation

The widespread adoption of the Internet, combined with its inherent insecurities, has led to the creation of numerous protocols with the intention of hiding data from the eyes of malicious observers. Unfortunately, the creation of such protocols has proved to be a non-trivial task. Security protocols must be designed with extreme care to avoid flaws that can result in total compromise of the protected data. These

flaws may not be immediately obvious, and in some cases have been discovered after a protocol has been in use for a number of years, such as the attack presented by Denning and Sacco (1981).

With the development of formal frameworks that treat the interaction of processes via communication channels, such as Hoare's CSP (Hoare, 1985) and Milner's π-calculus (Milner, 1999), it has become possible to represent these protocols formally and thus rigorously reason about their properties. This allows security protocols to be proved secure against known attacks.

It should be emphasised, however, that despite reference to security "proofs", a formal methods approach can never prove the total security of a protocol. Any proof relies on a model or abstraction of the protocol as implemented, and makes assumptions that can become potential points of attack. What these proofs provide is a much higher level of assurance of the correctness of the protocol.

By formally defining anonymity as a kind of security property, it should be possible to construct proofs of the anonymity provided by systems in a similar fashion to the approach taken towards proving traditional security properties, such as authentication or secrecy.

There are a number of different approaches towards proving security protocols. The formal approaches towards security began with the use of formal logics. The logic approach has been gradually superseded by process calculi approaches, and their accompanying use of automated tools to prove properties of systems. These major approaches are reviewed here.

7.2 Logics

Formal (symbolic) logics grew out of a desire to form a mathematical model of reasoning, which could be used to express the logical relationship between stated concepts, and to generate new "true" statements by the application of rules to existing statements. Propositions are proved using these rules, from facts that are already known and basic axioms that are assumed to be true.

The underlying rules differ between the various formal logics, depending on the scope and purpose of the logic in question. Different logics express notions of belief, knowledge, uncertainty, or even ignorance, within specific domains.

The application of formal logics to the analysis of security protocols was one of the first approaches taken towards the verification of such systems, with the most notable example being the BAN logic (Burrows, Abadi, and Needham, 1989). Logics have been shown to detect a range of problems with protocols whilst being reasonably easy to use. However, logics are a high level abstraction of a system, and may allow flaws that exist in the lower-level protocol implementation to pass undetected.

Here, we review the major forms of symbolic logic that relevant to security proofs. In general, these are based on modal logic (see section 7.2.2).

7.2.1 Some basic notation

In representing statements for logical manipulation, certain basic notation is commonly used:

p	:	The statement of a proposition p
$p \wedge q$	:	conjunction: "p and q". If p and q are both true then $p \wedge q$ is also true.
$p \vee q$	:	disjunction: "p or q". If either or both of p and q are true, then $p \vee q$ is also true.
$\neg p$	:	negation: "not p". In some logics, $\neg\neg p$ is not necessarily provably equal to p. In some logics, negation is not present.
$p \rightarrow q$	:	implication: "p implies q". This is equivalent to $\neg p \vee q$.
$p \leftrightarrow q$	:	equivalence: "p if and only if q". This is equivalent to $(p \rightarrow q) \wedge (q \rightarrow p)$.

Modal logics use all the tautologies of propositional logic, and generally also rely on the rule of "modus ponens":

If p and $p \rightarrow q$, then q

7.2.2 Modal Logics

Modal logics consider questions of necessity and possibility. The logics of this family are concerned with qualifiers that concern the state, or *modality*, of propositions based on sets of defining axioms. The basic syntactic elements, or "modalities", are the two statements that represent possibility and necessity:

$\Diamond p$	:	it is possible that p
$\Box p$	:	it is necessary that p

These may each be expressed in terms of the other, using negation:

- $\Diamond p$, "it is possible that p" $\leftrightarrow \neg\Box\neg p$ or "it is not necessary that not p"
- $\Box p$, "it is necessary that p", $\leftrightarrow \neg\Diamond\neg p$ or "it is not possible that not p"

There are many forms of modal logic that rely on different sets of axioms. The most common axiom set is modal logic S5, originally proposed by Lewis (1918):

1. $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$
2. $\Box p \rightarrow p$
3. $\Diamond p \rightarrow \Box \Diamond p$

The first of these axioms describes the distribution property of the necessitation operator, that if it is necessary that p implies q then if it is necessary that p it is also necessary that q. The second axiom simply expresses that if it is necessary that p then p is true. The third axiom states that if it is possible that p, then it is necessary that it is possible that p.

These $S5$ axioms allow a wide range of expressive power, and provide a basis for more advanced forms of modal logic. Numerous other sets of axioms also exist.

One semantics for modal logics is given in terms of *possible worlds*. Propositions may be true in some worlds but not in others (for example, the proposition "it is raining"). Necessity can then be modelled as propositions that must be true in all worlds.

In treating anonymity systems, we are concerned with modalities such as those expressed above. For systems of agents as observed by an attacker attempting to compromise anonymity, we are interested in the possibility that any member of a set of agents could have performed an action. The attacker will seek to reduce this to the smallest number of possible originators, whilst it is the aim of the system to prevent this. Modal concepts may prove useful in constructing a useful definition of anonymity for more advanced models.

Modal logic in its strict sense refers to the two basic statements of possibility and necessity, but the name is also used to refer to other logics that use different modalities. We now review those other logics that are relevant for anonymity models.

7.2.3 Epistemic Logics

Epistemic logics are concerned with propositions of knowledge, uncertainty, and ignorance. Knowledge, in the sense used by these logics, refers to an agent's justified beliefs based on observed facts (as opposed to *doxastic* logics, which treat beliefs of an agent based on lesser levels of justification).

Logics of knowledge add operators to express the knowledge held by a particular agent:

$K_i\ p$: agent i knows (has justified belief in) proposition p

K_i may be considered a necessity operator similar to $\Box$ in the previous section. We therefore have a corresponding possibility of falsity:

$\neg K_i \neg p$: agent i does not know that p is not true.

When we consider that an agent may know a particular proposition, then the basic axioms that express reasoning about this knowledge may be given, along with *modus ponens*, as:

- $K_i(p \rightarrow q) \rightarrow (K_i\ p \rightarrow K_i\ q)$
- $K_i\ p \rightarrow p$

which allow the deduction of facts from knowledge already possessed by the agent, and for an agent to deduce new facts based on observation. These correspond to the first and second axioms of S5 shown above, with K_i as the necessitation operator.

The capabilities of agents can vary greatly depending on the particular axioms of the epistemic logic. One important distinction is the quality of introspection, both positive and negative. This concept refers to the ability of an agent to know its own level of knowledge. The following axioms give positive and negative introspection:

- $K_i\ p \rightarrow K_i\ K_i\ p$ (Positive Introspection)
- $\neg K_i\ p \rightarrow K_i \neg K_i\ p$ (Negative Introspection)

With positive introspection, an agent i that knows a particular fact p also knows that it knows p. The second, and much stronger, case of negative introspection says that an agent is aware of its own ignorance: if i does not know p, then it knows that it does not know p. With the addition of these two axioms, the given system of epistemic logic closely resembles the modal logic S5 with K_i in place of the necessity operator.

Although not expressed in the context of anonymity systems, a comprehensive work on this approach is Halpern (1997), who considers several epistemic logics based on combinations of a small set of axioms. The work explores the possibilities and strengths of agents within these logical systems, and the inference rules that can be derived.

Epistemic logics consider the semantic "possible worlds" that can be constructed from the knowledge held within the system. Thus, if an agent knows a

fact p then it will not consider those worlds in which $\neg p$ is true. In expressing attacker models and the behaviour of agents, the knowledge that can be deduced by an agent from observed facts is of great importance to the anonymity that the system provides: we seek to prevent the revelation of facts that decrease the number of valid possible worlds.

7.2.4 Temporal Logics

Temporal logics add the dimension of time to propositions within a logic. This allows logics to express not only the truth of propositions, but also when the truth characteristic holds. This can add a significant level of complexity to a logic, however it increases the expressive power considerably. Although temporal logic covers all those logics that include statements concerning time, it is commonly used to refer to modal temporal logics similar to the *tense* logic introduced by Prior (1957). This approach considers timing as an ordering of events, but does not place any metric on that order.

There are four basic operators in Prior's temporal logic, split into two "weak" tense operators, and two "strong" tense operators:

- Weak
  $P\ p$: It has at some time been true that p.
  $F\ p$: It will at some time be true that p.
- Strong
  $H\ p$: It has always been the case that p.
  $G\ p$: It will always be the case that p.

Similar to the equivalence possible between necessity and possibility in modal logic, the weak tense operators may be expressed in terms of the strong tense operators by use of negation. For example, $P\ p \equiv \neg H \neg p$.

In the context of systems that aim to provide anonymity, and particularly in the context of censorship resistance, temporal logics are a way to express certain desirable properties. We may wish to prove that a certain fact concerning an agent is true at a particular moment, such as the possession of a certain amount of credit or resources. We may wish, however, to prevent this information from remaining known for longer than the course of the transaction. Temporal logics allow us to express the concept that a proposition may be true at a certain time, but not at other times.

There has been little research into using temporal logics to express anonymity, or even security, properties. This may be due to the complexity of temporal logics, combined with the ability to abstract away the temporal element of protocols. Few

existing protocols use explicit timing information, relying instead on single-use values, *nonces*, which mark that an event has taken place without reference to the time domain.

7.2.5 The BAN Logic

BAN (Burrows, Abadi, and Needham, 1989) is a modal logic designed to prove the correctness of authentication protocols. The logic allows the basic assumptions and goals of a protocol to be expressed as formulae in the syntax of the logic, along with the steps taken during the running of the protocol.

Deduction rules provide a reasoning path from the steps of the protocol to the desired goals. If this can be successfully achieved then the goals of the protocol are true.

BAN is based on the beliefs of participants, and some simple rules that allow the beliefs of an agent to be manipulated. BAN contains many constructs and deduction rules, including:

- $A \triangleleft X$: A *sees* X, or the message X has been sent to A.
- $A \mid\sim X$: A *said* X at some point.
- $A \models X$: A *believes*, or has justified belief in, X.
- $A \mapsto X$: A *has jurisdiction over* X.
- $\sharp(X)$: The message X is *fresh* (ie, it has not been seen before).
- $\{X\}_K$: The message X encrypted with the key K.

As mentioned earlier, many logics abstract away from timing information. BAN solves the problem of time by distinguishing two epochs, the past and present, by the use of the *fresh* construction $\sharp(X)$. This has proved sufficient to reason about a large number of complex protocols.

These constructions are manipulated using deduction rules that define the behaviour that results from such facts. For example, the rule of jurisdiction, which concerns an agent A's belief based on the perceived authority of another agent B, can be expressed in the following way:

$$\frac{A \models B \mapsto X,\ A \models B \models X}{A \models X}$$

which may be read as: "If A believes that B has authority over X and A believes that B believes X, then A believes X." There are many rules of this form (Burrows et al., 1989).

BAN has been applied to a number of well-known protocols, most famously in discovering a timing flaw in the Needham-Schroeder protocol that forms the basis of the widespread Kerberos authentication mechanism. (That BAN discovered this is unsurprising, as the logic was developed in response to the discovery of this flaw by other means.)

There have been a number of logics over the years that have sought to expand the capabilities of the BAN logic, such as GNY (Gong, Needham, and Yahalom, 1990) and SVO (Syverson and van Oorschot, 1994). These have yet to gain the level of acceptance enjoyed by BAN, and the increasing use of process calculi and automated theorem provers has caused approaches such as these to be less examined than previously.

BAN, as a logic of authentication, may provide an interesting insight into what is required of anonymity protocols, that can be seen as requiring the opposite of authentication. We wish to believe that a party with whom we are conversing is unknown to us, and as such cannot hold any of the properties that the BAN logic seeks to prove. Despite the fall of such logics from favour in recent years, the concepts that they embody are useful to us at a high level of abstraction.

7.3 Process Calculi

Process calculi provide a mathematical notation for describing communicating processes. They take the view of computers as communicating entities in larger networks, rather than stand-alone machines. This focus on communicating processes makes them ideal for the expression of anonymity systems that are by their very definition concerned with the communication between entities.

7.3.1 CSP

CSP (Communicating Sequential Processes) was developed by Hoare as a method for describing communicating processes operating in parallel:

> "... input and output are basic primitives of programming and that parallel composition of communicating sequential processes is a fundamental program structuring method." Hoare (1978)

CSP was originally described by Hoare (1978), and then updated (Hoare, 1985). Schneider (1999) provides a general introduction.

CSP has gained a wide following in the formal methods community, and has undergone many interesting developments, such as timing (Reed and Roscoe, 1986), and probabilistic elements (Morgan et al., 1996). There is mature tool support for CSP in the form of the model checker FDR (Formal Systems, 2004).

CSP has been applied to the analysis of security protocols, which are intuitively applicable to a calculus based on the interaction between communicating parties Ryan and Schneider (2001). It has been applied to a basic analysis of dining cryptographer networks, in Schneider and Sidiropoulos (1996) (see section 6.2).

CSP processes are defined in a syntax that allows the basic constructs of a standard programming language, including choice operators and logical expressions. A brief introduction to the syntax of CSP is given here.

A process definition may be given as a sequence of events. Events may be combined with a process using the *prefixing operator* $\rightarrow$; so $a \rightarrow P$ performs the event a then behaves like process P, and $a \rightarrow b \rightarrow P$ performs the event a then the event b and then behaves like process P.

Processes can be recursive:

$$\begin{aligned} Loop &= a \rightarrow b \rightarrow Loop \\ &= \mu P.a \rightarrow b \rightarrow P \end{aligned}$$

The second of these, labelled with μP, allows for a recursive definition without the need to name the process, and may be used in other process expressions without prior definition.

CSP has sequential execution of processes: $P\,;\,Q$

The operations that may be performed on processes are:

- Basic Operations

$P(n)$	:	Process P parameterized with value n.
$?x : E \rightarrow P(x)$	:	Engage in any event $x \in P$, then behave like $P(x)$.
$A \Box B$	:	Deterministically choose between the initial events of P and Q, and then behave accordingly.
$b \& P$	:	(For a boolean b) if b then enable P else stop.

- Parallel Composition

$P \vert\vert Q$	:	Parallel composition of P and Q requiring full synchronization of events.
$P \vert\vert_X Q$	:	Parallel composition of P and Q requiring synchronization on the events of a set X.
$P \vert\vert\vert Q$	:	Parallel composition of P and Q with no synchronization.
$P \backslash Q$	:	Hide events from the set Q from outside observers.
Pa/b	:	Rename all variables a in P to b.

- Primitive Processes

  $Stop$: Deadlocked process.
  $Skip$: Successfully terminating process.

CSP is concerned with the sets of observations that can be made of a process. The simplest useful form of observations is the *trace*, the sequence of events that the process engages in. More complex observations include *failures*, *divergences*, and *refusals*, which contain extra information about the state of the system and expand ability to reason about a process. By examining these observations it is possible to prove the adherence of a model to a stated set of goals.

Schneider and Sidiropoulos (1996) use CSP to model and prove anonymity properties of the dining cryptographer network (section 6.2) consisting of three participants, assuming a reliable broadcast channel. The model is constructed to prove the assertion that any data within the system could have originated from any actor within the system: any permutation of the sequence of events occurring within a run of the model results in the observations of the system.

Although this proof considers only a toy anonymity system, based on the impractical dining cryptographer network, and does not consider an active attacker with the ability to inject or block messages, the techniques used provide inspiration for further work into proving anonymity properties with process calculi. The proof is also one of the very few extant examples of a formal methods proof of anonymity in the literature.

7.3.2 The π-Calculus

The π-calculus is a fundamental reworking of the ideas expressed in Milner (1980)'s calculus of communicating systems (CCS). CCS and CSP describe communicating processes in a similar fashion and offer the same level of expressive power.

In seeking to describe communicating processes as they exist in the networked world that has emerged since the development of CCS and CSP, however, Milner observes that:

> "Physical systems tend to have permanent physical links; they have *fixed* structure. But most systems in the informatic world are not physical; their links may be virtual or symbolic... These symbolic links can be created or destroyed on the fly..." — Milner (1999):

The π-calculus extends the basic capabilities of CCS to include *mobility*: agents can form new links with other agents, and can destroy old links. An agent may therefore begin life in one area of a system and, in the course of execution, relocate to an entirely new portion of a system.

Processes send and receive messages along defined channels. In the π-calculus these messages may include the name of a channel. This powerful addition allows the dynamic creation of new topologies in the system. The basic structure of the calculus is presented here as described by Milner (1999).

There are an infinite set $\mathcal{N}$ of *names*, and a corresponding disjoint set $\overline{\mathcal{N}}$ of *co-names*. There is a set of *variables* disjoint from names and co-names that may be instantiated to arbitrary values. These structures are collectively known as *terms*.

Terms

The set of terms is defined (where M, N are terms) by:

n	:	name (shown as lowercase to distinguish from generic terms)
x	:	variable
(M, N)	:	pair

Process Expressions

The set of *action prefixes*, π, corresponds to the sending or receiving of a name, along with the silent transition τ. These are represented by:

$x(y)$	Accept y on channel named x
$\overline{x}\langle y\rangle$	Send y along channel named x
τ	Unobservable action

Processes can be formed as follows:

- $\sum_{i\in I} \pi_i.P_i$: *summation* of processes, corresponding to the execution of an action (sending or receiving a name, or the internal transition τ), followed by some process P_i (where I is a finite indexing set)
- $(P_1|P_2)$: *composition* of P_1 with P_2, which allows P_1 and P_2 to run concurrently
- $(\nu a)P$: *restriction* of the name a to process P, resulting in a being bound in P
- $!P$: *replication* of process P, which allows (infinite) parallel repetition

This set represents the most basic form of the π-calculus. Names may be passed along channels, processes have the ability to run both sequentially and in parallel, replication can be expressed, and the scope of names may be restricted to processes using the ν operator.

The π-calculus has spawned variants designed for the analysis of differing interacting systems. Of interest for the treatment of security protocols is that of Abadi and Gordon (1997) which adds cryptographic primitives to the calculus. That paper, and further work (Abadi, 1999; Gordon and Jeffries, 2004), provides a framework, in addition to methods and tools, for the rigorous analysis of security protocols.

Work in proving security protocols using the π-calculus and its variants often uses "observational equivalences" between processes. This compares a concrete model of the protocol, and an abstract specification that expresses the security properties. Using the calculus, an equivalence is proven between the concrete model of the protocol and the abstract properties.

There has been work on developing an executable language based on the calculus. This includes Pict (Pict, 2004), and an extended version with increased mobile agent communication capabilities called Nomadic Pict (Pict, 2004). The occam-pi project (Occam-Pi, 2004) combines the CSP basis of the occam programming language with the mobility of the π-calculus. The existence of languages in which π-calculus models can more easily be expressed could potentially increase the utility of the calculus for obtaining experimental results.

There does not yet appear to have been any work on expressing anonymity systems within the framework of the π-calculus. The flexibility offered by the calculus seems to be ideal for representing many of the network topologies that are being used by the latest anonymity systems, and the existing body of work into security proofs using the π-calculus provide a source of techniques that could be of use in proving anonymity properties.

7.3.3 Comparisons

The two approaches described above do not by any means form an exhaustive list of existing process calculi. There are many other process calculi in existence, such as elements of the ISO standard LOTOS (ISO, 1987) for formal description of systems, Bergstra's Algebra of Communicating Processes with Abstraction (ACP) (Bergstra and Klop, 1985) and many variants based on the π-calculus (Parrow and Victor, 1998; Abadi and Gordon, 1997; Herescu and Palamidessi, 2000). The multitude of π-calculus variants and CSP are, however, the most widely used and studied process calculi, certainly in formal methods security research.

The fundamental difference between the π-calculus and CSP is in the approach taken toward semantics. CSP uses denotational semantics to treat the set of observations that can be made of processes. The π-calculus, and its predecessor CCS, use algebraic semantics to prove equivalences in systems.

CSP has been a well-established language for many years, and is supported by a wide variety of mature tools to aid in proofs (Formal Systems, 2004). There is

a large body of work both in using and extending the language, including work on security proofs (Lowe, 1996; Ryan and Schneider, 2001) and the previously mentioned work on anonymity (Schneider and Sidiropoulos, 1996).

Additionally, there has been work into transforming the abstract language of CSP into executable forms. The parallel programming language occam, originally designed for the INMOS transputer, is strongly based on the version of CSP presented in Hoare (1978), and there has been work into transforming CSP specifications into other executable languages (Stepney, 2003; Gardner, 2003). The ability to create models that can feasibly be executed greatly increases the potential for real-world experimentation in addition to abstract proofs.

The π-calculus, however, has itself received some attempts towards an executable form. As is mentioned above, both Pict (Pict, 2004) and Nomadic Pict (Pict, 2004) are strongly based on the π-calculus, although these projects are less developed than their CSP counterparts.

The main strength of the π-calculus over CSP lies in its explicit ability to model mobility. In the π-calculus, channel names may be passed as data in order to create new links between agents in the system. In attempts to model the interactions between anonymous participants, and also to express the relationships between different aspects of user identities, the ability to model the creation and destruction of links in the system could be of great use.

Both CSP and the π-calculus have been extended in many ways. Of importance to modelling anonymity systems will be factors such as the inclusion of cryptographic operations (Abadi and Gordon, 1997), the potential for asynchronous communications (Herescu and Palamidessi, 2000), and the addition of probabilistic capabilities (Herescu and Palamidessi, 2000; Lowe, 1995). Any of these may be found in either calculus, however the explicit expression of mobility possible in the π-calculus may turn out to be an important advantage over CSP, despite the latter's extensive tool support.

Bibliography

Martín Abadi. Secrecy by typing in security protocols. *Journal of the ACM*, 46 (5):749–786, 1999.

Martín Abadi and Andrew D. Gordon. A calculus for cryptographic protocols: The spi calculus. In *Fourth ACM Conference on Computer and Communications Security*, pages 36–47. ACM Press, 1997.

Ross Anderson. The eternity service. In *Proceedings of Pragocrypt '96*, 1996.

APA-S. `alt.privacy.anon-server` usenet newsgroup, 2004.

Adam Back, Ulf Möller, and Anton Stiglic. Traffic analysis attacks and trade-offs in anonymity providing systems. In Ira S. Moskowitz, editor, *Proceedings of Information Hiding Workshop (IH 2001)*, pages 245–257. LNCS 2137, Springer, April 2001.

J. A. Bergstra and J. W. Klop. Algebra of communicating processes with abstraction. *Theoretical Computer Science*, 37(1):77–121, May 1985.

Oliver Berthold and Heinrich Langos. Dummy traffic against long term intersection attacks. In Dingledine and Syverson (2002a).

Oliver Berthold, Andreas Pfitzmann, and Ronny Standtke. The disadvantages of free MIX routes and how to overcome them. In Federrath (2000).

Louis Brandeis and Samuel Warren. The right to privacy. In *Harvard Law Review*, volume 4, December 1890.

Michael Burrows, Martín Abadi, and Roger Needham. A logic of authentication, from proceedings of the royal society, volume 426, number 1871, 1989. In *William Stallings, Practical Cryptography for Data Internetworks, IEEE Computer Society Press, 1996*. IEEE Computer Society Press, 1989.

David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 4(2), February 1981.

David Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1:65–75, 1988.

Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In Federrath (2000), pages 46–66.

Richard Clayton. Improving onion notation. In Dingledine (2003).

Thomas McIntyre Cooley. *A Treatise on the Law of Torts*. Callaghan, 2 edition, 1888.

Cypherpunks. Cypherpunks mailing list. `http://www.csua.berkeley.edu/cypherpunks/Home.html`, 2004.

Wei Dai. Pipenet 1.1. `http://www.eskimo.com/~weidai/pipenet.txt`, August 1996.

George Danezis. Mix-networks with restricted routes. In Dingledine (2003).

George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, May 2003.

George Danezis and Len Sassaman. Heartbeat traffic to counter (n-1) attacks. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2003)*, Washington, DC, USA, October 2003.

Chrysanthos Dellarocas. Analyzing the economic efficiency of ebay-like online reputation reporting mechanisms. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 171–179. ACM Press, 2001.

Dorothy Denning and Giovanni Sacco. Timestamps in key distributed protocols. *Communications of the ACM*, 24(8):533–535, 1981.

Claudia Diaz and Bart Preneel. Reasoning about the anonymity provided by pool mixes that generate dummy traffic. In *Proceedings of the 6th Information Hiding Workshop, IH 2004*. LNCS, Springer, May 2004a.

Claudia Diaz and Bart Preneel. Taxonomy of mixes and dummy traffic. Presented at the 3rd Working Conference on Privacy and Anonymity in Networked and Distributed Systems, August 2004b.

Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In Dingledine and Syverson (2002a).

Roger Dingledine, editor. *Proceedings of Privacy Enhancing Technologies workshop (PET 2003)*, March 2003. LNCS 2760, Springer.

Roger Dingledine, Michael J. Freedman, David Hopwood, and David Molnar. A Reputation System to Increase MIX-net Reliability. In Ira S. Moskowitz, editor, *Proceedings of Information Hiding Workshop (IH 2001)*, pages 126–141. Springer-Verlag, LNCS 2137, April 2001.

Roger Dingledine, Michael J. Freedman, and David Molnar. The free haven project: Distributed anonymous storage service. In Federrath (2000).

Roger Dingledine, Nick Mathewson, and Paul Syverson. Reputation in P2P Anonymity Systems. In *Proceedings of Workshop on Economics of Peer-to-Peer Systems*, June 2003.

Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004a.

Roger Dingledine, Vitaly Shmatikov, and Paul Syverson. Synchronous batching: From cascades to free routes. `http://freehaven.net/doc/sync-batching/sync-batching.pdf`, 2004b.

Roger Dingledine and Paul Syverson, editors. *Proceedings of Privacy Enhancing Technologies workshop (PET 2002)*, April 2002a. LNCS 2482, Springer.

Roger Dingledine and Paul Syverson. Reliable MIX Cascade Networks through Reputation. In Matt Blaze, editor, *Proceedings of Financial Cryptography (FC '02)*. LNCS 2357, Springer, March 2002b.

eBay, 2004. eBay. `http://www.ebay.com`, 2004.

Dirk Engelmann and Urs Fischbacher. Indirect reciprocity and strategic reputation building in an experimental helping game. `http://citeseer.ist.psu.edu/600045.html`, November 2002.

European Union. European union directive on data protection, 1998.

H. Federrath, editor. *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, July 2000. LNCS 2009, Springer.

Uriel Feige, Amos Fiat, and Adi Shamir. Zero knowledge proofs of identity. In *Proceedings of the 19th ACM Symp. on Theory of Computing*, pages 210–217, May 1987.

Formal Systems. Formal systems homepage. `http://www.fsel.com/software.html`, 2004.

FreeNet, 2004. The freenet project homepage. `http://freenet.sourceforge.net`, 2004.

W.B. Gardner. Bridging csp and c++ with selective formalism and executable specifications. In *First ACM & IEEE International Conference on Formal Methods and Models for Codesign (MEMOCODE'2003)*. ACM, June 2003.

Ian Goldberg. *A Pseudonymous Communications Infrastructure for the Internet*. PhD thesis, UC Berkeley, December 2000.

Ian Goldberg and David Wagner. TAZ servers and the rewebber network: Enabling anonymous publishing on the world wide web. *First Monday*, 3(4), August 1998.

David M. Goldschlag, Michael G. Reed, and Paul F. Syverson. Hiding routing information. In R. Anderson, editor, *Proceedings of the 1st International Workshop on Information Hiding*, pages 137–150. LNCS 1174, Springer, 1996.

Li Gong, Roger Needham, and Raphael Yahalom. Reasoning About Belief in Cryptographic Protocols. In Deborah Cooper and Teresa Lunt, editors, *Proceedings 1990 IEEE Symposium on Research in Security and Privacy*, pages 234–248. IEEE Computer Society, 1990.

Andrew Gordon and Martin Jeffries. Cryptyc: A tool for type-checking security protocols. `http://research.microsoft.com/~adg/cryptyc.htm`, 2004.

J. Halpern and K. O'Neill. Secrecy in multiagent systems. In *Proc. 15th IEEE Computer Security Foundations Workshop*, pages 32–46, 2002.

Joseph Y. Halpern. A theory of knowledge and ignorance for many agents. *Journal of Logic and Computation*, 7(1):79–108, 1997.

Joseph Y. Halpern and Kevin R. O'Neill. Anonymity and information hiding in multiagent systems. *Journal of Computer Security*, 2004.

Oltea Mihaela Herescu and Catuscia Palamidessi. Probabilistic asynchronous pi-calculus. In *Foundations of Software Science and Computation Structure*, pages 146–160, 2000.

HMSO. Data protection act. HMSO, 1984.

HMSO. Data protection act. HMSO, 1998. `http://www.hmso.gov.uk/acts/acts1998/19980029.htm`.

C. A. R. Hoare. Communicating sequential processes. *Comm. ACM*, 21(8):666–677, August 1978.

C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.

Dominic Hughes and Vitaly Shmatikov. Information hiding, anonymity and privacy: A modular approach. *Journal of Computer Security*, 12(1):3–36, 2004.

ICCPR. International covenant on civil and political rights. `http://www.hrweb.org/legal/`, January 1997.

ISO. ISO 8807: Information processing systems – open systems interconnection – LOTOS – a formal description technique based on the temporal ordering of observational behaviour. Standard, International Standards Organization, Geneva, Switzerland, 15 February 1987. First edition.

JAP. Jap – anonymity and privacy. `http://anon.inf.tu-dresden.de/index_en.html`, 2004.

JAP Ruling, 2004. Ruling on storage of JAP data. `http://www.datenschutzzentrum.de/material/themen/presse/anonip3_e.htm`, 2004.

Dogan Kesdogan, Jan Egner, and Roland Büschkes. Stop-and-go MIXes: Providing probabilistic anonymity in an open system. In *Proceedings of Information Hiding Workshop (IH 1998)*. LNCS 1525, Springer, 1998.

Clarence Lewis. *A Survey of Symbolic Logic*. University of California Press, 1918. Republished by Dover, 1960.

Gavin Lowe. Probabilistic and prioritized models of timed CSP. In *Selected papers of the meeting on Mathematical foundations of programming semantics*, pages 315–352. Elsevier, 1995.

Gavin Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 1055, pages 147–166. Springer, 1996.

Robin Milner. *A Calculus of Communicating Systems*. LNCS 92, Springer, 1980.

Robin Milner. *Communicating and mobile systems: the π calculus*. Cambridge University Press, 1999.

MixMaster, 2004. Mixmaster project development page. http://mixmaster.sourceforge.net, 2004.

Ulf Möller, Lance Cottrell, Peter Palfrader, and Len Sassaman. Mixmaster Protocol — Version 2. http://www.abditum.com/mixmaster-spec.txt, July 2003.

Carroll Morgan, Annabelle McIver, Karen Seidel, and J. W. Sanders. Refinement-oriented probability for CSP. *Formal Aspects of Computing*, 8(6):617–647, 1996.

Occam-Pi, 2004. occam-pi: blending the best of CSP and the pi-calculus. http://www.cs.kent.ac.uk/projects/ofa/kroc/, 2004.

Joachim Parrow and Bjorn Victor. The fusion calculus: Expressiveness and symmetry in mobile processes. In *Logic in Computer Science*, pages 176–185, 1998.

Andreas Pfitzmann and Marit Köhntopp. Anonymity, unobservability, and pseudonymity: A proposal for terminology. http://freehaven.net/anonbib/papers/Anon_Terminology_v0.14.pdf, July 2000.

Andreas Pfitzmann and Michael Waidner. Networks without user observability – design options. In *Proceedings of EUROCRYPT 1985*. LNCS 219, Springer, 1985.

Pict. The Pict Programming Language. http://www.cis.upenn.edu/~bcpierce/papers/pict/Html/Pict.html, 2004.

Pict, 2004. The Nomadic Pict Page. http://www.cl.cam.ac.uk/users/pes20/nomadicpict.html, 2004.

Richard Posner. The economics of privacy. In *American Economic Review Papers and Proceedings*, 1981.

Jon Postel. RFC 793: Transmission control protocol. http://www.ietf.org/rfc/rfc793.txt, September 1981.

A. N. Prior. *Time and Modality*. Oxford, 1957.

J.-J. Quisquater, L. Guillou, and T. Berson. How to explain zero-knowledge protocols to your children. In G. Brassard, editor, *Theory and Applications of Cryptology, CRYPTO'89, Santa Barbara, USA*, pages 628–631. LNCS 435, Springer, 1990.

Jean-François Raymond. Traffic Analysis: Protocols, Attacks, Design Issues, and Open Problems. In Federrath (2000), pages 10–29.

G. M. Reed and A. W. Roscoe. A timed model for communicating sequential processes. In *International Colloquium on Automata, Languages and Programming on Automata, languages and programming*, pages 314–323. Springer, 1986. ISBN 0-387-16761-7.

Michael Reiter and Aviel Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1), June 1998.

Marc Rennhard and Bernhard Plattner. Practical Anonymity for the Masses with Mix-Networks. In *Proceedings of the IEEE 8th Intl. Workshop on Enterprise Security (WET ICE 2003)*, Linz, Austria, June 2003.

Peter Ryan and Steve Schneider. *Modelling and analysis of security protocols*. Addison-Wesley, 2001.

Steve Schneider. *Concurrent and Real Time Systems: The CSP Approach*. John Wiley, 1999.

Steve Schneider and Abraham Sidiropoulos. CSP and anonymity. In *ESORICS*, pages 198–218, 1996.

Andrei Serjantov and George Danezis. Towards an information theoretic metric for anonymity. In Dingledine and Syverson (2002a).

Andrei Serjantov, Roger Dingledine, and Paul Syverson. From a trickle to a flood: Active attacks on several mix types. In Fabien Petitcolas, editor, *Proceedings of Information Hiding Workshop (IH 2002)*. LNCS 2578, Springer, October 2002.

Andrei Serjantov and Richard E. Newman. On the anonymity of timed pool mixes. In *Proceedings of the Workshop on Privacy and Anonymity Issues in Networked and Distributed Systems*, pages 427–434, Athens, Greece, May 2003. Kluwer.

Claude E. Shannon. A mathematical theory of communication. *Bell System Tech. J.*, 27(3):379–423, July 1948. Continued 27(4):623-656, October 1948.

Vitaly Shmatikov. Probabilistic model checking of an anonymity system. *Journal of Computer Security*, forthcoming.

Susan Stepney. CSP/FDR2 to Handel-C translation. Technical Report YCS-2002-357, Department of Computer Science, University of York, June 2003.

P. Syverson and P. van Oorschot. On unifying some cryptographic protocol logics. In *Proc. IEEE Symposium on Research in Security and Privacy*, pages 14–28, 1994.

Paul Syverson, Gene Tsudik, Michael Reed, and Carl Landwehr. Towards an Analysis of Onion Routing Security. In Federrath (2000), pages 96–114.

Paul F. Syverson and Stuart G. Stubblebine. Group principals and the formalization of anonymity. In *World Congress on Formal Methods (1)*, pages 814–833, 1999.

United Nations. Universal Declaration of Human Rights, 1948.

United States Department of Commerce. United States Department of Commerce Safe Harbor Program, 2004.

Michael Waidner and Birgit Pfitzmann. The dining cryptographers in the disco: Unconditional sender and recipient untraceablility with computationally secure serviceability. In J.-J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology—EUROCRYPT 89, Belgium*, page 690. LNCS 434, Springer, 1990.

Marc Waldman, Aviel D. Rubin, and Lorrie Faith Cranor. Publius: A robust, tamper-evident, censorship-resistant, web publishing system. In *Proc. 9th USENIX Security Symposium*, pages 59–72, August 2000.