Lecture Notes in Computer Science 2760
Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

Springer
*Berlin*
*Heidelberg*
*New York*
*Hong Kong*
*London*
*Milan*
*Paris*
*Tokyo*

Roger Dingledine (Ed.)

# Privacy Enhancing Technologies

Third International Workshop, PET 2003
Dresden, Germany, March 26-28, 2003
Revised Papers

Springer

Volume Editor

Roger Dingledine
The Free Haven Project
Cambridge, MA, USA
E-mail: arma@mit.edu

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress.

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at <http://dnb.ddb.de>.

CR Subject Classification (1998): E.3, C.2, D.4.6, K.6.5, K.4, H.3, H.4, I.7

ISSN 0302-9743
ISBN 3-540-20610-8 Springer-Verlag Berlin Heidelberg New York



Typesetting: Camera-ready by author, data conversion by Olgun Computergrafik
Printed on acid-free paper SPIN: 10972243 06/3142 5 4 3 2 1 0

# Preface

PET 2003 was the 3rd Workshop on Privacy Enhancing Technologies. It all started in 2000 with Hannes Federrath organizing "Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability," July 25–26, 2000, held at the Computer Science Institute (ICSI), Berkeley, CA (LNCS 2009). Roger Dingledine, Adam Shostack, and Paul Syverson continued in April 2002 in San Francisco (PET 2002, LNCS 2482). This year was Dresden, and as long as the new PET field prospers, we intend to hold this workshop annually.

The workshop focused on the design and realization of anonymity and anti-censorship services for the Internet and other communication networks. Besides the excellent technical papers, we had four panels, led by Richard Clayton, Andrei Serjantov, Marit Hansen, and Allan Friedman. This year we also extended our work-in-progress talk schedule, allowing 24 people from the audience to introduce a variety of new technologies and perspectives.

An event like PET 2003 cannot happen without the work and dedication of many individuals. First we thank the authors, who wrote and submitted 52 full papers. Next the program committee, who wrote 163 reviews and selected 14 papers for presentation and publication, with additional reviewing help from Peter Berlich, Oliver Berthold, Steve Bishop, Jan Camenisch, Sebastian Clauß, Allison Clayton, George Danezis, Christian Friberg, Philippe Golle, Mike Gurski, Guenter Karjoth, Dogan Kesdogan, Stefan Köpsell, Thomas Kriegelstein, Heinrich Langos, Nick Mathewson, Richard E. Newman, Richard Owens, David Parkes, Peter Pietzuch, Sandra Steinbrecher, Nathalie Weiler, Matthew Wright, and Sheng Zhong.

Besides this scientific work, organizational issues had to be taken care of: Martina Gersonde and Sandra Steinbrecher did a great job in handling all issues relating to the local administration. Stefan Köpsell and Silvia Labuschke gave all kinds of technical support, including providing WLAN Internet access at the workshop facilities. We are grateful to Secunet for providing us with free Internet access.

We tried to keep costs to a minimum but also offer many diverse social activities. The core business was paid for by the registration fees completely. In addition, we received generous sponsorship from Microsoft Europe and from MITACS, making it possible to offer stipends to students and other researchers so they could attend PET 2003. These contributions really helped us to bring together all parts of the community so we could push the field forward.

August 2003

Roger Dingledine and Andreas Pfitzmann
PET 2003 Chairs

# Privacy Enhancing Technologies 2003
# Dresden, Germany
# March 26–28, 2003

## Program Committee

Alessandro Acquisti, SIMS, UC Berkeley, USA
Stefan Brands, Credentica, Canada
Jean Camp, Kennedy School, Harvard University, USA
David Chaum, USA
Richard Clayton, University of Cambridge, UK
Lorrie Cranor, AT&T Labs Research, USA
Roger Dingledine, The Free Haven Project, USA (Program Chair)
Hannes Federrath, Freie Universitaet Berlin, Germany
Ian Goldberg, Zero Knowledge Systems, Canada
Marit Hansen, Independent Centre for Privacy Protection, Germany
Markus Jakobsson, RSA Laboratories, USA
Brian Levine, University of Massachusetts at Amherst, USA
David Martin, University of Massachusetts at Lowell, USA
Andreas Pfitzmann, Dresden University of Technology, Germany
Matthias Schunter, IBM Zurich Research Lab, Switzerland
Andrei Serjantov, University of Cambridge, UK
Adam Shostack, Canada
Paul Syverson, Naval Research Lab, USA

## General Chair

Andreas Pfitzmann, Dresden University of Technology, Germany

## Sponsors

Microsoft Europe
MITACS

# Table of Contents

# Mix-Networks with Restricted Routes

George Danezis

University of Cambridge, Computer Laboratory,
William Gates Building, 15 JJ Thomson Avenue,
Cambridge CB3 0FD, United Kingdom.
George.Danezis@cl.cam.ac.uk

**Abstract.** We present a mix network topology that is based on sparse expander graphs, with each mix only communicating with a few neighbouring others. We analyse the anonymity such networks provide, and compare it with fully connected mix networks and mix cascades. We prove that such a topology is efficient since it only requires the route length of messages to be relatively small in comparison with the number of mixes to achieve maximal anonymity. Additionally mixes can resist intersection attacks while their batch size, that is directly linked to the latency of the network, remains constant. A worked example of a network is also presented to illustrate how these results can be applied to create secure mix networks in practise.

**Keywords:** Mix networks, mix cascades, traffic analysis, anonymity.

## 1 Introduction

Mix networks were introduced by Chaum [4] as a technique to provide anonymous communications. The messages to be anonymized are relayed by a sequence of trusted intermediaries, called mixes, to make the task of tracing them through the network as difficult as possible. Nested layers of encryption and strict length restrictions are additionally used to make the inputs of each mix node bitwise unlinkable to its outputs.

Further research into mix networks, has been divided between real time systems, primarily for web browsing, such as onion routing [12], webmixes [1] or the freedom network [3], and non real-time systems such as babel [13], mixmaster [18] and the newer mixminion [6]. Other issues have been the trade off between real time guarantees and anonymity properties, proper metrics to quantify anonymity [24,7], and the importance of cover traffic to maintain anonymity.

In this paper we present and discuss some proposals about the topology that mix networks might assume. These are on one hand a fully connected graph, on the other a mix cascade. We then discuss the advantages and disadvantages of a restricted network topology, that can be modeled as a network corresponding to a sparse constant degree graph, and analyze it using existing work on expander graphs. Finally we compare the anonymity and other properties provided by this new topology against the more traditional ones.

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 1–17, 2003.

We prove that such restricted networks scale well in the number of mix nodes. The route length necessary to provide maximal anonymity grows only logarithmically in the number of nodes in the network, and the total amount of genuine traffic required to protect the network against traffic analysis and intersection attacks grows linearly with the number of mix nodes.

The paper is organized in the following fashion: in section 2 we present previous work on mix network topologies namely fully connected mix networks and cascades. We then proceed in section 3 to introduce a new network topology based on expander graphs and give a brief summary of its advantages. In section 4 we introduce a framework for analyzing mix networks and definitions of possible attacks. Then in section 5 we analyze the properties of the new expander graph topology introduced, and in particular the route lengths necessary, the volumes of traffic to resist intersection attacks and its resilience to corrupt nodes. In section 6 we compare the new topology with an analysis of fully connected mix networks and cascades. We then illustrate our results, and how they can be used in practice by creating and studying an example network in section 7. Finally we present some possible avenues for future work in section 8.

## 2 Previous Work

Some work has already been done on the topology of mix networks. The network topology influences how clients choose the path their messages take through the mix network. The original proposal by Chaum [4] assumes a fully connected graph, while mix cascades [14,20,2] force a particular sequence of mixes to be used. The freedom network [3] only allows restricted routes to be used, for performance reasons but without any published analysis about what repercussions on anonymity such restrictions on the network might have. In [2] Berthold *et al* briefly introduces the possibility of having mix networks that are sparse, but then as we will see, focuses on describing the benefits of mix cascades.

### 2.1 General Mix Networks

In [4] David Chaum introduces mix networks as a collection of nodes that relay messages between each other from their original senders to their final recipients. Throughout the paper there is an implicit assumption that all nodes can communicate with all other nodes, therefore forming a fully connected network. Clients choose the path their messages take by selecting a sequence of nodes at random, from the set of all existing mix nodes. Mixmaster [18] which follows quite closely Chaum's original proposals, also allows clients to choose any route through the network, using reliability statistics [17] to select nodes. Other proposals concerning route selection use reputation metrics [8] to quantify how reliable mixes in a network are.

### 2.2 Mix Cascades

While the fully connected nature of the mix networks seemed to improve the anonymity provided, Berthold *et al* [2] found that they can be vulnerable against

very powerful adversaries, such as those who control all the mix nodes except one. In particular if only one of the mixes in the network is honest the anonymity of the messages going through it will most likely be compromised. Attackers can perform intersection attacks, while the probability that all nodes in a short path are compromised is very high. Additionally if two or more messages follow the same route, attacks are trivial to perform.

As a solution a *cascade* of mixes is proposed. Users of the network are not free to choose which route to take, but are all forced to route their messages though a predefined sequence of mixes. Further work has been done to improve the reliability of networks of cascades against corrupt nodes using reputation [9]. The obvious drawbacks of cascades are the small anonymity sets they provide in the general case, and the fact that they do not scale well to handle heavy load. Cascades are also vulnerable to denial of service attacks, since disabling one node in the cascade will stop the functioning of the whole system. Some solutions are proposed to solve the problem that active attacks could pose, but require user authentication to work properly [2].

## 3 Mix Networks Based on Expander Graphs

We propose a mix network with a network topology based upon sparse, constant degree graphs, where users may only choose routes following this topology. Furthermore each link out of a node should be chosen according to a predefined probability distribution. Therefore selecting a path in the network can be approximated as a random walk on the corresponding weighted graph. We will show that this network provides some of the advantages of cascades, while being more scalable. We will provide theoretical anonymity bounds using the proposed metric for anonymity based on entropy in [24], and define under which conditions the network provides anonymity close to the theoretical limit. Minimum traffic bounds to prevent the network being vulnerable to traffic analysis and intersection attacks are also calculated.

The topology that we propose for the mix network is based on expander graphs. Expanders are a special family of graphs with the following properties: a D-regular graph is a $(\mathcal{K}, \mathcal{A})$-expander if for every subset $\mathcal{S}$ of vertexes of $\mathcal{G}$, if $|\mathcal{S}| \leq \mathcal{K}$, then $|N(\mathcal{S})| > \mathcal{A}|\mathcal{S}|$ where $|\mathcal{S}|$ is the number of vertexes in $\mathcal{S}$ and $|N(\mathcal{S})|$ is the number of nodes sharing an edge (neighbouring) with a vertex in $\mathcal{S}$. In plain language it means that any random subset of nodes will have "many" different neighbouring nodes. In practise expanders have a relatively small and constant degree $D$ in relation to the number of edges of the graph, and a large expansion factor $A$, that is proportional to the number of "neighbours". A good introduction to expander graphs and their applications can be found in [16].

A relevant result is that most bipartite graphs with degree at least three provide good expansion properties, which means that a topology based on a random bipartite graph with each mix node having three fixed neighbors would be an expander with high probability [22]. Therefore such networks can be constructed by brute force, or by using the surveyed or proposed methods in [23].

The families of expanders with explicit constructions presented have a constant, but large, degree but also an arbitrary large number of nodes, which makes them practical for large networks.

The first question that comes to mind is quantifying the anonymity that such networks provide in comparison to fully connected networks. In a fully connected network a message coming out of the network has a probability of originating initially from a particular node proportional to the input load of the node. As we will see a random walk though the expander graph will converge toward the same probability after a number of steps proportional to $\mathcal{O}(\log N)$ where $N$ is the number of nodes in the network [10]. This represents the *a-priori* knowledge of an adversary that only knows the topology of the graph, but no details about the actual traffic going through it.

Intersection attacks presented in [2] rely on the fact that messages using the same sequence of nodes will only occur in a fully connected network with a very small probability. Since a mix network based on a small constant degree graph only provides a limited choice of routes for messages to take, a node can wait so that enough messages are accumulated before sending them, to make sure that all its neighbors always receive messages. Because there is only a linear number of routes to fill with dummy traffic, only order $\mathcal{O}(DN)$ messages are required where $N$ is the number of nodes and $D$ the degree of the graph. This strategy is more efficient than filling all the $\mathcal{O}(N^2)$ links in a fully connected graph, since adding more nodes only requires the total traffic to increase linearly in order to maintain the network's resistance to traffic analysis.

Before we move on to prove the properties described above, as we will do in section 5, we will first introduce a way of quantifying anonymity and some definitions about the attacks that can be performed on mix networks.

## 4 A Framework for Analyzing Mix Networks

In order to compare fully connected mix networks, mix cascades and restricted routing there is a need to have a way of quantifying not only their security but also their efficiency. Efficiency can be measured following the usual paradigms of communication networks, namely the latency of messages and the load on mix servers. On the other hand security, and in particular anonymity, does not have a well established way of being measured.

In order to quantify the anonymity provided by a network of mixes we will use the metric proposed by Serjantov and Danezis [24]. We will consider the sender anonymity set of a message as the entropy of the probability distribution describing the likelihood particular participants were senders of the message. As expected the anonymity of messages increases as the number of potential senders increases. Given a certain number of participants, the anonymity of a message is also maximized when all participants have an equal probability of having been the sender of a message. Recipient anonymity can be quantified in an equivalent fashion, so we will only present the analysis of sender anonymous properties.

A message $m_e$ exits the mix network at time $t_e$ from node $n_e$. The network is made out of $N$ mix nodes, $n_1$ to $n_N$. Messages $m_{ij}$ are injected at node $n_i$ at time $t_j$. The task of an attacker is to link the message $m_e$ with a message $m_{ij}$.

We consider the probability distribution

$$\begin{aligned} p_{ij} &= \Pr[m_e \text{ is } m_{ij}] \\ &= \Pr[m_e \text{ is } m_{ij} | m_e \text{ inserted at } n_i] \times \Pr[m_e \text{ inserted at } n_i] \end{aligned} \tag{1}$$

that describes how likely the input messages in the network are to have been message $m_e$. We can express this probability as the probability that a node $n_i$ was used to inject a message, multiplied by the probability a particular message $m_{ij}$ injected at this node is $m_e$. The entropy of the probability distribution $p_i$ is the effective sender anonymity set of the message. Because of the strong additive property of entropy we can calculate this entropy as:

$$\begin{aligned} \mathcal{A} &= H(p_{ij}) \\ &= H(\Pr[m_e \text{ inserted at } n_i]) \\ &+ \underbrace{\sum_{x \in 1 \ldots N} \Pr[m_e \text{ inserted at } n_x]}_{\text{traffic analysis attacks}} \times \underbrace{H(\Pr[m_e \text{ is } m_{ij} | m_e \text{ inserted at } n_x])}_{\text{traffic confirmation attacks}} \end{aligned} \tag{2}$$

An attacker might attempt to reduce the anonymity by subjecting the network to traffic analysis in order to reduce the uncertainty of $\Pr[m_e \text{ inserted at } n_i]$. We shall therefore name $\mathcal{A}_{\text{network}} = H(\Pr[m_e \text{ inserted at } n_i])$, the anonymity provided by the network. This quantifies how effectively the traffic injected to or ejected from particular nodes is mixed with traffic from other nodes. Given a particular threat model if no technique is available for the attacker to reduce the uncertainty of $\mathcal{A}_{\text{network}}$ beyond her a-priori knowledge, we can say that *the network is resistant to traffic analysis in respect to that particular threat model.*

The attacker can also try to reduce the anonymity set of the message by reducing the uncertainty of the probability distribution describing the traffic introduced at the mix nodes, $\Pr[m_e \text{ is } m_{ij} | m_e \text{ inserted at } n_x]$. The attacker can do this by using additional information about $m_e$ and $m_{ji}$, like the times $t_e$ the message comes out of the network or $t_j$ the time it was injected in the network. She can also do this by flooding nodes, or stopping messages arriving to the initial nodes. It is worth noting that a network might protect users from traffic analysis, but still provide inadequate anonymity because of such side information leaked by messages as they enter and exit the mix network. Side information is not limited to time, but can also be associated with the protocol or mechanism used, client type, unique identifiers or route length indications observed at the edges of the mix network. Attacks that try to link messages using such side information leaked at the edges of the network, instead of tracing the message through the network, are called *traffic confirmation attacks* [26].

In analyzing and comparing the anonymity provided by networks with restricted routes we will limit ourselves into considering the traffic analysis resistance since it depends heavily on the topology while traffic confirmation attacks

depend on the particular mix batching and flushing strategy individual nodes use. Having defined a way of quantifying the anonymity provided by the network, we will study in the next section, the route length necessary to archive maximal anonymity in expander graph based mix networks and the volumes of traffic necessary to avoid traffic analysis attacks.

## 5 Anonymity Analysis of Restricted Network Topologies

In a fully connected mix network it is intuitive that a message that comes out of a mix node, after a number of hops, could have originated from any node in the network with a probability proportional to its input load. Since users chose their initial nodes at random, or taking into account in the case of mixmaster reliability statistics [17], we can say that the probability the messages originated from an initial node is equal to the probability a client has to choose this node as an entry point to the network. The same probability distribution is often used to determine the intermediate and final node of the anonymous path. This observation allows us to compute $\mathcal{A}_{\text{network}}$ for fully connected networks, using the probability distribution describing the selection of the entry node.

For a graph that is not fully connected we need to calculate what the probability is that a message that is present in a node after a number of mixing steps has originated from a particular initial node. This requires us to trace the message backwards in the network. If the graph is not directed the likelihood a message was injected at a particular node is equal to the probability a random walk starting at the final node finishes on a particular node after a certain number of hops.

Therefore, we consider the network as a graph and the act of selecting a path through it as a random walk, and we model the route selection procedure and actual communication as a Markov process. In practice some anonymous route selection algorithms exclude nodes from being present on the path of a message twice, which violates the memoryless property of the Markov process. Despite this if we assume that a Markov process is still a good approximation to the route selection process, after an infinite number of steps the probability a message is present on a particular node should be equal to the stationary probability distribution $\pi$ of the process. Therefore the maximum anonymity provided by the network will be equal to its entropy, $\mathcal{A}_{\text{network}} = H(\pi)$

For reasons of efficiency we need to find how quickly the probability distribution $q^{(t)}$ describing where a message is after a number of random steps $t$, converges to the stationary probability $\pi$ of the Markov process. A smaller $t$ would allows us to minimize the number of hops messages need to be relayed for, therefore reducing the latency and increasing the reliability of the network. Motwani and Raghavan [19] provide a theoretical bound on how quickly a random walk on a graph converges to the stationary probability. If $\pi_i$ is the stationary distribution of a random walk on a graph $G$ and $q^{(t)}$ the probability distribution after $t$ number of steps starting from any initial distribution $q^{(0)}$. We define $\Delta(t)$ as the relative point wise distance as follows:

$$\Delta(t) = \max_i \frac{|q_i^{(t)} - \pi_i|}{\pi_i} \tag{3}$$

It can be shown [19] that this distance after a number of random steps $t$ is bound by the number of nodes in the network $N$ and the second eigenvalue $\lambda_2$ of the transition matrix corresponding to the graph of the network:

$$\Delta(t) \leq \frac{\sqrt{N}(\lambda_2)^t}{\min_i \pi_i} \tag{4}$$

Therefore the distance decreases exponentially as the number of step $t$, for which the message travels through the networks, increases linearly.

It is clear that the quick rate of convergence of the probability distribution is dependent on the second eigenvalue being small. An extensive body of research has concentrated on linking the value of the second eigenvalue to expansion properties of graphs, to show that good expanders exhibit small second eigenvalues (see [19] for details). There is a fundamental limit of how quickly a network can mix that depends on the degree $D$ of the graph:

$$\lambda_2 \geq \frac{2\sqrt{D-1}}{D} \tag{5}$$

The results above assure us that a mix network with a topology corresponding to a good expander graph would mix well, in a number of steps logarithmic in its size, $\mathcal{O}(\log N)$. This means that in this small number of steps a message that enters the network will leave the network at a node selected with probability approaching the probability after an infinite number of steps, namely the stationary probability distribution $\pi$. Furthermore its degree could be bound in order to allow for links to be padded with cover traffic, to protect against intersection attacks or traffic analysis attacks, as we will study in the next section.

In fact the methods described above can be used to calculate the theoretical probability that a messages that comes out at a mode $n_e$ of the network has been injected at another node $n_i$. In theory the a-priori knowledge of the attacker, concerning where a message was injected, corresponds to the probability distributions after the random walk on the graph representing the network. It also depends on the way that initial nodes are being chosen by clients, using reliability statistics, or other information. As the number of intermediaries grows this distribution converges towards being the same for all initial choices of entry nodes. Therefore as the number of hops grow a network based on expander graphs offers *uniform anonymity*, which means that the anonymity provided by the network is the same regardless of the node used to inject a message. In the next section we will study how much traffic is needed to make the network resistant to traffic analysis, in other words an actual observation of its running will not give the attacker any additional information beyond the theoretical calculations presented above.

A number of ways can be employed in order to find an expander graph that would represent a good anonymous communication network. If the number of

nodes is small it can be done by brute force, until a graph is found with a second eigenvalue that approaches the limit described above. Explicit constructions employing Ramanujan graphs [11] or zig zag products [23] can also be employed to construct the network. Standard graphs such as multi dimensional hyper-cubes also exhibit properties that could be suitable.

### 5.1 Protection Against Intersection Attacks

An advantage of mix cascades, as argued in [2], is that they are not susceptible to intersection attacks. Such attacks use the fact that many messages are sent using the same path, to perform traffic analysis attacks and follow the messages through the network. The authors note that, if every time a message is sent by the user under surveillance, the set of possible destinations of every mix is intersected with the set of possible destinations of previous messages, then the actual path of the message will become apparent. This is due to the very small probability the same, even partial, route is used by different messages. Since in mix cascades all messages use the same sequence of intermediary nodes, such an attack does not yield any results. Of course traffic confirmation is always possible, by observing all the edges of the network, and find correlations between initial senders and final recipients. Such attacks will always be possible if the network does not provide full unobservability [21], or other specific countermeasures.

In a mix network with restricted routes and small degree, such as one based on expander graphs described in the previous section, the potential for intersection attacks described above, can be greatly reduced. This is done by making sure that all the links from a node to its neighbors are used in a flushing cycle. This is possible in practice since the number of these links in the network is relatively small, and does not grow proportionally to $\mathcal{O}(N^2)$ as for fully connected networks. Making sure that all links are used is sufficient to foil the simplest intersection attacks, that use the intersection of sets of potential senders to trace messages [15]. Traffic analysis is still possible if the probability a messages has used a link is skewed. Therefore we need enough genuine traffic to be mixed together for the observable load on the network links to be proportional to the theoretical probability distribution described by the transition matrix.

Using a threshold mix as an example we will calculate how much traffic is needed for no link of a node to be left empty. We assume that clients select the next hop from a particular node using a probability distribution $p_n$, where $n$ is the number of $N_i$ neighboring nodes. Then the probability that the link to a node is left empty in a batch of $b$ messages is:

$$\Pr[\exists i.N_i \text{ empty}] < \Pr[N_1 \text{ empty}] + \ldots + \Pr[N_n \text{ empty}] \tag{6}$$

$$\Pr[\exists i.N_i \text{ empty}] < \sum_{\forall N_i} (1 - p_i)^b \tag{7}$$

As the size of the batch of messages to be processed grows, the probability that a link is empty decreases exponentially, making simple intersection attacks infeasible. It is important to note that the same effect can be achieved by adding

dummy traffic on the links that are not used. Again the amount of dummy traffic in the network will only grow linearly with the number of nodes in the network.

In order to avoid the attacker gaining any more information than the theoretical anonymity, which is the entropy of the stationary probability distribution on the nodes $E_{\pi_i}$, the actual flows of messages on the links should be as close as possible to the matrix describing the network topology. As described above each node receives a number of messages $b$, some of which will be output on a particular link $i$ according to a binomial distribution, with probability $p_i$. We can require the number of messages that are actually transmitted not to diverge on a particular round or time period by more than a small percentage $f$ from the average mean. As the number of messages $b$ received by the mix increases the probability that $X$ the number of messages transmitted on the link $i$, is close to the expected mean $bp_i$ increases:

$$\Pr[(1-f)bp_i \leq X \leq (1+f)bp_i] = 1 - 2\Phi\left(-fk^{\frac{1}{2}}\sqrt{\frac{p_i}{1-p_i}}\right) \tag{8}$$

Where $\Phi$ is the cumulative probability distribution of a normal random variable, with mean zero and variance one. We can require $f$ to be arbitrary small, like .05, by mixing more messages together in a threshold mix [25]. Expressing the above formula to calculate $f$ makes it clear that the deviation from the mean expected traffic gets smaller proportionally to the inverse square root of the number of messages processed. This result can then be used in conjunction with $p_{min}$, the probability associated with the link that is least likely to be used in the network or mix, to derive how much genuine traffic would be necessary in a node to protect against traffic analysis.

Another way of calculating the appropriate threshold value for a threshold mix would be to calculate the number of rounds necessary to perform the intersection attack. The techniques used do do this are related to the statistical disclosures attacks described in [5,15]. The attacker performs a hypothesis test on each of the links, with $H_0$ representing the hypothesis that the stream of messages under surveillance are output on the link under observation, and $H_1$ representing the hypothesis the messages are output on another link. In case $H_0$ is true the volume of messages on the observed link follows a probability distribution $Y_0 = k + X_{b-1}$ otherwise it follows a probability distribution $Y_1 = X_{b-1}$, where $b$ is the threshold of the mix, $k$ the number of mixing rounds, and $p_i$ the probability the link is used by messages. $X_{b-1}$ is the random variable following the binomial distribution with probability $p_i$ after $b-1$ trials. The mean and standard deviation of these distributions are:

$$\mu_{Y_0} = k + k(b-1)p_i \qquad \sigma^2_{Y_0} = k(b-1)p_i(1-p_i) \tag{9}$$

$$\mu_{Y_1} = k(b-1)p_i \qquad \sigma^2_{Y_1} = k(b-1)p_i(1-p_i) \tag{10}$$

In order to be able to accept or reject hypothesis $H_0$ we will require the observed volume of traffic to be within a distance of a few standard deviations $\sigma_{Y_0}$ from the mean $\mu_{Y_0}$, while also at a minimum distance of a few standard deviations $\sigma_{Y_1}$ from $\mu_{Y_1}$ to avoid false positives. The number $l$ of standard deviation depends

on the degree of confidence required. The minimum number of mixing rounds $k$ that need to be observed by an attacker to confirm of reject the hypothesis can therefore be calculated by:

$$\mu_{Y_0} - l\sigma_{Y_0} > \mu_{Y_1} + l\sigma_{Y_1} \tag{11}$$

$$k > 4l^2 \frac{p_i}{1-p_i}(b-1) \tag{12}$$

For values of $l = 1$ we get a confidence of 68%, for $l = 2$, 95% and for $l = 3$, 99%. The above formula is true both for general mix networks and for mix networks with restricted routes. We can require the value of rounds $k$ to be greater than one $k > 1$, with $l = 0.6745$ for the attacker to have only a confidence of 50% in order to frustrate traffic analysis of messages that are not part of a stream that follows the same route.

### 5.2 Corrupt Nodes

An important factor that has to be taken into account when judging an anonymous network, is how robust it is to corrupt nodes. In particular one has to assess the likelihood that all the nodes that have been selected to be on the path of a message are corrupt nodes. For the topology presented this amount to determining the probability $p_{l/c}$ that $l$ nodes selected by a random walk on the expander graph, might include $c \leq l$ corrupt nodes. Gillman provides an upper bound for this probability [10], that is dependent on the expansion properties of the graph, and the "probability mass" of the corrupt nodes.

If the matrix representing the graph of the mix network has a second eigenvalue $\lambda_2$ then define $\epsilon = 1 - \lambda_2$. Assume that the set $C$ of nodes is corrupt. Then define $\pi_c$ as the probability mass represented by this corrupt set, $\pi_c = \sum_{i \in C} \pi_i$ where $\pi$ is the stationary probability distribution of the random walk on the graph. After a number of steps $l$ the probability that a walk has only been performed on corrupt nodes is:

$$\Pr[t_c = l] \leq \left(1 + \frac{(1-\pi_c)\epsilon}{10}\right) e^{-l\frac{(1-\pi_c)^2\epsilon}{20}} \tag{13}$$

The probability that a path is totally controlled by corrupt nodes therefore depends on the amount of traffic processed by the corrupt nodes, and the mixing properties of the graph, but decreases exponentially as the route length increases. Assuming a particular threat model the route length can be increased until that threat is very improbable. In practice the constant factors of the bound are too large to lead to values of the route length that are practical in real systems. Therefore despite the initially encouraging results, for even modest $\pi_c$ other methods might have to be used to determine and minimize the probability that the full route is composed of corrupt nodes.

## 6 Comparing Sparse Networks with Other Topologies

We have studied in the previous sections some properties of sparse mix networks namely their necessary route length, batch size or volume of traffic necessary to provide nearly maximal anonymity. We shall next compare these properties with previously introduced topologies.

Sparse networks based on expander graphs scale well by providing maximal network anonymity for a route length $l$ proportional to $\mathcal{O}(\log N)$. Furthermore they can be made resistant to traffic analysis and intersection attacks using a constant volume of traffic per node, depending on the degree $D$ of the network. By (12) we observe that if the route selection algorithm is uniform, then the batch size $b$ of nodes can be $b < \frac{1}{4l^2}k(D-1)+1$ which is independent of the number of nodes in the network.

### 6.1 Mix Cascades

Given our definitions it is clear that a mix cascade is resistant to traffic analysis, since observing the network traffic does not provide an attacker with more information than she originally had about the correspondence of input to output nodes. This is the case because there is no uncertainty about the node where all messages were inserted, since there is only one. The fact that $\mathcal{A}_{\text{network}} = 0$ does not mean that the network does not provide any anonymity to messages, but simply that all the anonymity provided to the messages originates conceptually form the single $H(\Pr[m_e \text{ is } m_{ij} | m_e \text{ inserted at } n_x])$ component of (2).

This absolute protection against traffic analysis comes at a very high cost. The anonymity provided is reduced to the volume of messages that can be processed by the node with least throughput in the cascade. The latency of the messages is also large, since each message has to be processed by all nodes in the cascade.

Despite the inefficiencies presented above mix cascades are a valuable design. They are resistant to very powerful adversaries, that control all nodes but one. They also highlight the advantages of implementing topologies that can be analyzed, in order to understand their anonymity properties.

### 6.2 Mix Networks

General mix networks are distinct from sparse, constant degree, mix networks because senders of anonymous messages are allowed to follow arbitrary routes through them. This sometime is misinterpreted as meaning that matrix corresponding to the mix network is fully connected. Indeed an attacker that has no additional knowledge of the network, beyond the way routes are selected, has no other way of attributing probabilities linking output messages to input nodes, other than by using a random walk on this fully connected graph, for a number of steps corresponding to the route length.

An attacker that can observe the traffic in the network, on the other hand, can get much better results. If we assume that the number of nodes is larger

than the threshold of the mixes, some links remain unused in each mix round. Furthermore even if the threshold is comparable to the number of mixes, the volume of messages sent will give the attacker a different probability distribution from the theoretical one described by the route selection distribution. Therefore an attacker can use the additional information, extracted from these observation to trace messages more effectively.

We will denote the graph used for the route selection through the network as $G$. This graph has $N$ nodes, the number of mixes, that are all connected with each other by weighted edges. The weights correspond to the probability that a node is selected as the next mix in the path, and can be uniform if the selection is performed at random, or it can be based on reliability statistics or reputation metrics. Given a column vector $v$ describing where a message was injected, the probability $P$ a messages comes out of the network at a particular nodes after $l$ steps, can be calculated to be $P_l = G^l v$ This is the *a-priori* information that an attacker has about the correspondence between input and output nodes, even before any traffic analysis has been performed.

As the attacker observes the network, for round $i$ it can deduce a matrix $G_i$ with the mixes as the vertexes, and the traffic load that was observed between them during round $i$ as the weights on the edges. It is worth observing that $G_i$ is closely related to $G$ in the sense that the selection of routes for any round is performed using $G$, but is sparse if the threshold of the mixes is lower than the number of nodes. In fact the weights on the edges follow the same probability distribution as for $G$, but are going to be different subject the the variance of the multinomial distribution, and the threshold of the mixes. An adversary that observes the actual traffic patterns in the networks will therefore be able to have more accurate information about where the messages injected are going, by calculating the probability distribution $P'_l = G_l \dots G_2 G_1 v$.

The relation of $G_i$ the graph of the traffic observed at round $i$ with the graph $G$ used to route messages, is crucial in understanding the anonymity that generic mix networks provide. The smaller the difference between $G_i$ and $G$ the more resistant the network will be to traffic analysis. In order for $G_i$ to be close to $G$ there needs to be enough traffic to make the mean load on all the links proportional to the probabilities of the route selection, as described for sparse topologies in section 4.2. In general one would expect $\lim_{l\to\infty} H(P'_l) = \lim_{l\to\infty} H(P_l)$, but also $\forall l, H(P_l) \leq H(P'_l)$, where $H(\cdot)$ denotes the entropy of a distribution.

For $G_i$ to be therefore a good approximation of $G$ it is necessary each round to fill all links with traffic volumes proportional to the values on the edges of $G$. This requires the volumes of traffic handled by the network to be proportional to $\mathcal{O}(N^2)$ as the number of nodes $N$ in the network grows. The batch size that needs to be handled by each node therefore grows proportionally in the size of the network $b < \frac{k}{4l^2}(N-1)+1$, as described by (12). The increased batch sizes also has a repercussion on the latency of messages that travel through the network since mixes will wait for more messages before they operate.

Mix networks that do not use a deterministic threshold mixing strategy, where the first batch of messages to go in are also the first batch of messages to go out, can also be analyzed in a similar fashion by redefining $G_i$. It would then need to represent the probability distributions leading to the effective anonymity sets of messages instead of the volumes of traffic in the network.

## 7 An Example Network: Putting It All together

In order to illustrate how all the results presented on mix networks based on expander graphs fit together, we will present an example network and analyze it. We will proceed to calculate the route length necessary for it to provide uniform anonymity, the amount of real traffic that should be present in each node for it to be resistant to traffic analysis and intersection attacks.

### 7.1 Selecting a Good Topology

We aim to create a network with $N = 400$ mix nodes, each with $D = 40$ neighbors. The neighbor of a mix both sends and receives messages from the mix and therefore we can represent this network as an undirected graph. Furthermore we will assume that senders will choose their path across the network using a random walk on the graph, with equal weights on all the edges. Therefore the probability that a messages follows a particular link, given that it is already on a node is equal to $p_n = p_{min} = \frac{1}{40}$.

Using a brute force algorithm we create a number of networks and compute their second eigenvalue until a network with good expansion properties is found. After testing less than ten candidates we find a graph with a second eigenvalue $\lambda_2 = 0.3171$, which is close to the theoretical limit given by equation (5) of $\lambda_2 > 0.3122$

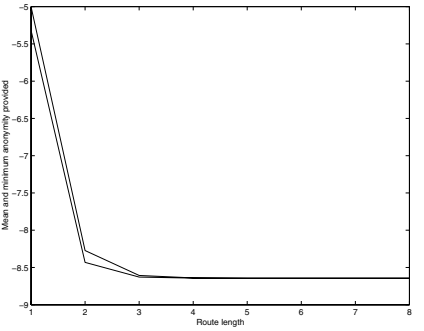
As expected such a graph has $N_l = 16 \cdot 10^3$ links instead of $N^2 = 16 \cdot 10^4$ that a fully connected graph would have. Therefore it is sparse in the sense that only one in ten links are used.

### 7.2 Mixing Speed

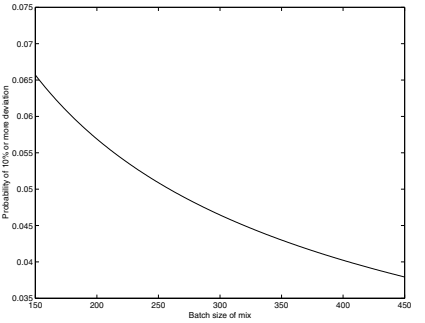
Using the theoretical formula (4) we know that the network will provide nearly uniform anonymity after a number of mixing steps proportional to $\log N$. From the graph we know that the $\min_i \pi_i = \frac{1}{400}$ since the stationary distribution is uniform, and therefore the theoretical anonymity, according to [24], should be equal to $\mathcal{A} = -log_2 N = -8.6438$.

In theory the relative point wise distance $\Delta(t)$ between the observed $q^{(t)}$ distribution after $t$ steps and the stationary distribution $\pi_i$ should converge following $\Delta(t) \leq n\sqrt{n}\lambda_2^t$. This allows us to calculate that the safe route length is around six. In practice much tighter bounds can be computed by directly calculating using $G^t$ the distributions $q^{(t)}$ from the available graph $G$. It is therefore observed that after four steps of the random walk the entropy of $q^{(t)}$ is equal

to the theoretical entropy calculated above. Figure a illustrates this by showing how the mean entropy provided to messages entering on any node compares with the minimum entropy that is offered by the network. Their convergence indicates that after four steps the network provides uniform and also maximum anonymity.




(a) Mean and lowest entropy after a random walk




(b) Probability the distribution deviates more than 10%

### 7.3 Resisting Intersection and Traffic Analysis Attacks

In order to avoid the simplest forms of intersection attacks all the networks links need to be used for every round. The probability a network link is not used is described by equation (7). For this particular network all $p_i = \frac{1}{40}$ where $p_i$ is the probability a link is followed. The probability that any link is left empty for threshold mix with batch size $b = 300$ is therefore, $\Pr[\exists i.N_i \text{ empty}] < 2\%$. Therefore for batches larger than 300 messages the probability a link is left empty is very small.

In order to protect against more sophisticated traffic analysis attacks taking into account statistical differences in the observed distributions from the graph $G$, we need to calculate the probability this deviation is large (for example larger than 10% as shown in figure b). In practice with a batch size of $b = 300$ as specified above, the attacker would need to observe $k > 4\frac{1}{40-1}(300-1) = 30$ messages in a stream in order to have a confidence of 68% that a particular link was used.

## 8 Future Work

An area that has not been investigated in depth has been the creation of the graph topology. Since the routes are restricted there is a need to advertise the allowed routes to clients, but also for the mixes to collaboratively decide upon a topology. If a brute force algorithm is used some randomness about the initial

seed could be contributed by each mix so that the result is assured not to be biased. If an explicit construction is employed a similar procedure should be used to make sure that the parameters of the network are not set by a minority of potentially corrupt nodes, as discussed in [9].

Besides good mixing properties, expander graphs provide some useful robustness properties against deletion of nodes. A major concern when building a network is the number of nodes an adversary needs to disable necessary to partition the network, or reduce the anonymity it provides. Assessing the impact of removing nodes on the speed of mixing would be a good start for assessing this risk.

Finally strategies for countering active flooding or delaying attacks are necessary. Since the number of neighboring nodes is small, they are more likely than in a fully connected network to all be corrupt and mount active attacks against the surrounded honest nodes.

## 9 Conclusions

The case has been argued in this paper that sparse networks provide desirable properties against traffic analysis attacks and scale better than fully connected networks or cascades. Some calculations presented, such as the probability a route is fully captured by adversaries, are theoretically appealing but do not provide tight enough bounds to be used in practice, while others are directly applicable for analyzing networks. Maybe tighter bounds could be found by restricting further the topology of the network.

The analysis of intersection attacks provides practical bounds to calculate the amount of traffic necessary to defend mixes, but is only applicable to threshold mixes. It is important to generalize it in the future to other mix batching and flushing strategies, such as pool mixes presented in [24,25]. It also offers a good foundation to decide how many messages in the stream are to travel in the same path. The required volume of data to make the network resistant to some rounds of traffic analysis can also be used as a guide to decide how much cover traffic is to be introduced.

The main contribution of this paper is that it highlights that a middle ground exists between free route mix networks, and extremely restrictive mix cascades. By designing the network carefully, and choosing appropriate topologies, some properties of both can be achieved, such as improved resistance to intersection attacks, along with shorter routes and better scalability.

## Acknowledgments

The author would like to thank Andrei Serjantov for the invaluable provocative discussions that greatly improved this work, and Richard Clayton for clearly formulating the problem surrounding intersection attacks.

## References


1. Oliver Berthold, Hannes Federrath, and Stefan Köpsell. Web MIXes: A system for anonymous and unobservable Internet access. In *Designing Privacy Enhancing Technologies, LNCS Vol. 2009*, pages 115–129. Springer-Verlag, 2000.
2. Oliver Berthold, Andreas Pfitzmann, and Ronny Standtke. The disadvantages of free MIX routes and how to overcome them. In *Designing Privacy Enhancing Technologies, LNCS Vol. 2009*, pages 30–45. Springer-Verlag, 2000.
3. P. Boucher, I. Goldberg, and A. Shostack. Freedom system 2.0 architecture. `http://www.freedom.net/info/whitepapers/`, December 2000. Zero-Knowledge Sytems, Inc.
4. David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 4(2), February 1981.
5. George Danezis. Statistical disclosure attacks. In *Sec 2003*, May 2003. `http://www.cl.cam.ac.uk/~gd216/StatDisclosure.pdf`.
6. George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *IEEE Security and Privacy Symposium*, 2003.
7. Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In *Privacy Enhancing Technologies Workshop 2002*, April 2002.
8. Roger Dingledine, Michael J. Freedman, David Hopwood, and David Molnar. A reputation system to increase MIX-net reliability. *Lecture Notes in Computer Science*, 2137:126–141, 2001.
9. Roger Dingledine and Paul Syverson. Reliable MIX Cascade Networks through Reputation. Proceedings of Financial Cryptography 2002.
10. David Gillman. A chernoff bound for random walks on expander graphs. In *IEEE Symposium on Foundations of Computer Science*, pages 680–691, 1993.
11. Yair Glasner. Ramanujan graphs with small girth.
12. D. Goldschlag, M. Reed, and P. Syverson. Onion routing for anonymous and private internet connections. *Communications of the ACM (USA)*, 42(2):39–41, 1999.
13. C. Gulcu and G. Tsudik. Mixing E-mail with Babel. In *Network and Distributed Security Symposium - NDSS '96*. IEEE, 1996.
14. Anja Jerichow, Jan Müller, Andreas Pfitzmann, Birgit Pfitzmann, and Michael Waidner. Real-Time MIXes: A Bandwidth-Efficient Anonymity Protocol. IEEE Journal on Selected Areas in Communications, 1998.
15. Dogan Kesdogan, Dakshi Agrawal, and Stefan Penz. Limits of anonymity in open environments. In *Information Hiding, 5th International Workshop*, Noordwijkerhout, The Netherlands, October 2002. Springer Verlag.
16. Nati Linial and Avi Wigderson. Expander graphs and their applications. Collection of Lecture Notes `http://www.math.ias.edu/~avi/TALKS/expander_course.pdf`, January 2003.
17. Christian Mock. Mixmaster stats (Austria). `http://www.tahina.priv.at/~cm/stats/mlist2.html`.
18. Ulf Möller and Lance Cottrell. Mixmaster Protocol–Version 2. Unfinished draft, January 2000. `http://www.eskimo.com/~rowdenw/crypt/Mix/draft-moeller-mixmaster2-protocol-00.txt`.
19. Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge Univerisity Press, 1995.

20. A. Pfitzmann, B. Pfitzmann, and M. Waidner. Isdnmixes: Untraceable communication with very small bandwidth overhead, 1991.
21. Andreas Pfitzmann and Marit Kohntopp. Anonymity, unobservability and pseudonymity — a proposal for terminology. In *Designing Privacy Enhancing Technologies: Proceedings of the International Workshop on the Design Issues in Anonymity and Observability*, number 2009 in LNCS, pages 1–9. Springer-Verlag, July 2000.
22. M. S. Pinsker. On the complexity of a concentrator. In *Proceedings of the 7th International Teletraffic Conference*, pages 318/1–318/4, Stockholm, 1973.
23. Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *IEEE Symposium on Foundations of Computer Science*, pages 3–13, 2000.
24. Andrei Serjantov and George Danezis. Towards an information theoretc metric for anonymity. In *Privacy Enhancing Technologies Workshop 2002*, San Francisco, CA, May 2002.
25. Andrei Serjantov, Roger Dingledine, and Paul Syverson. From a trickle to a flood: Active attacks on several mix types. In Fabien A.P. Petitcolas, editor, *Information Hiding Workshop*, number 2578 in LNCS, pages 36–52. Springer–Verlag, 2002.
26. P. Syverson, M. Reed, and D. Goldschlag. Private web browsing. *Journal of Computer Security*, 5(3):237–248, 1997.

# Generalising Mixes

Claudia Díaz[1] and Andrei Serjantov[2]

[1] K.U.Leuven ESAT-COSIC
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
claudia.diaz@esat.kuleuven.ac.be, http://www.esat.kuleuven.ac.be/cosic/
[2] University of Cambridge Computer Laboratory
Cambridge CB3 0FD, United Kingdom
Andrei.Serjantov@cl.cam.ac.uk, http://www.cl.cam.ac.uk/~aas23/

**Abstract.** In this paper we present a generalised framework for expressing batching strategies of a mix. First, we note that existing mixes can be represented as functions from the number of messages in the mix to the fraction of messages to be flushed.
We then show how to express existing mixes in the framework, and then suggest other mixes which arise out of that framework. We note that these cannot be expressed as pool mixes. In particular, we call *binomial mix* a timed pool mix that tosses coins and uses a probability function that depends on the number of messages inside the mix at the time of flushing. We discuss the properties of this mix.

## 1 Introduction

Many modern anonymity systems use the notion of a mix as introduced in [Cha81]. Chaum's original system used a very simple threshold mix, but over the last few years many different mixes have been proposed in the literature [Cot94,Jer00,KEB98].

One of the most important parameters of a mix is its *batching strategy*. Intuitively, the batching strategy of a mix is the algorithm for collecting the messages to be mixed together and forwarding them to the next hop. Naturally, this influences both the anonymity and message delay properties of the mix.

In the past the batching strategies of mixes were often described by giving the algorithm which determines when to flush the mix and how many (and which) messages to forward during the flush. In this paper, we present an simple formalism for describing mixes, which also enables a quick (qualitative) comparison. In the next section we show how existing mixes are described. In Section 4, we show that there are functions which express other mixes with interesting properties. We then focus on this mix, extend it and examine its properties.

## 2 Comparing Batching Strategies of Mixes

Let us examine existing mixes. There are several which we are familiar with from the literature (see survey in [SDS02]): threshold mix, timed mix, timed pool mix

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 18–31, 2003.

and the timed dynamic pool (Cottrell) mix[1]. We now seek to express mixes, just as an implementer would, as functions $P : \mathbb{N} \to [0, 1]$ from the number of messages inside the mix to the fraction of messages to be flushed. We now note that just this function is not enough to express the batching strategy of a mix. We also need to specify how often we would execute this function and flush messages. Note that in timed mixes, this is just amount to the period between mix flushes. The variable $n$ represents the number of messages contained in the mix at the time of flushing.

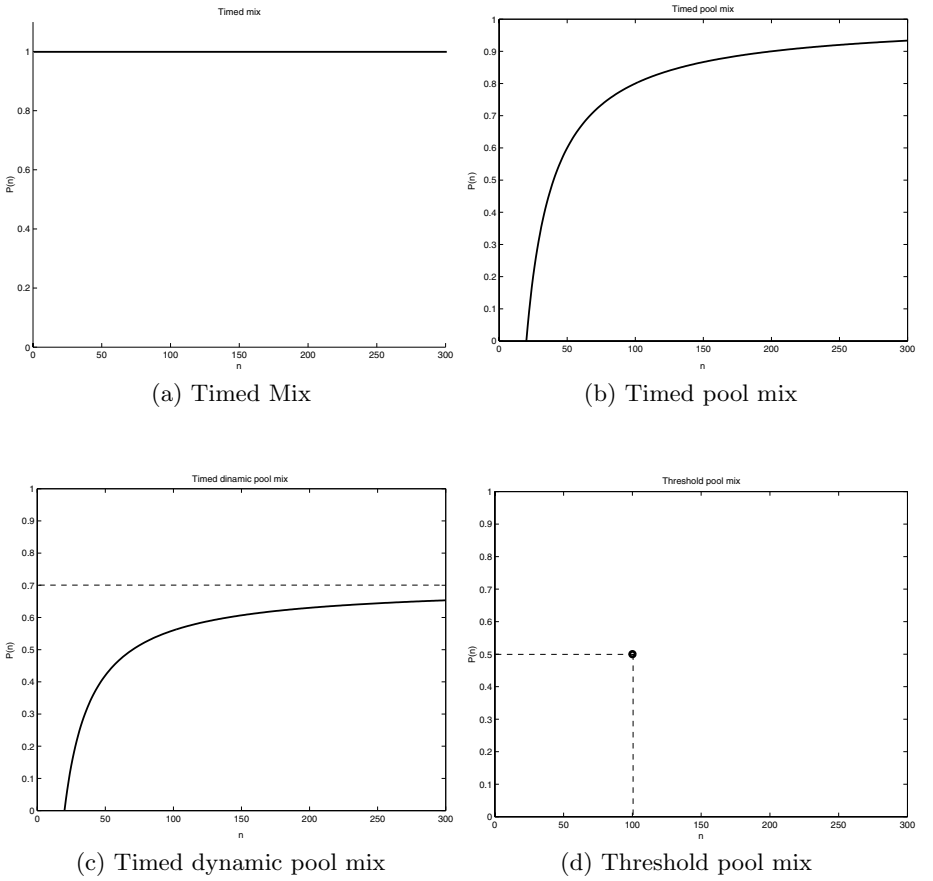
Figure 1 presents:

- Timed mix (a): This mix flushes all the messages it contains at the time of flushing. Therefore, the percentage of sent messages is always 100%, i.e., $P(n) = 1$.
- Timed pool mix (b): This mix keeps a constant number of messages, $N_p$, in the pool ($N_p = 20$ in this example), and flushes periodically . If the mix contains no more than $N_p$ messages at the time of flushing, it will not output any message. When it contains more, it outputs $n - N_p$ messages, that means that the percentage of sent messages can be expressed as: $P(n) = 1 - N_p/n$.
- Timed dynamic pool mix (Cottrell mix) (c): This mix outputs messages at the timeout only when the number of messages is greater than a threshold $N_p$. The number of output messages is a fraction, $f$, of the difference between the number of messages inside the mix and the value of the threshold of the pool, $f(n - N_p)$ ($f = 0.7$ and $N_p = 20$ in the example). In the figure, the function that represents the percentage of sent messages is $P(n) = f(1 - N_p/n)$.
- Threshold pool mix (d): We have noted above that each mix is a function, together with a time period ($T$) which specifies how often we flush the mix. If we set $T = 0$ and let the function $P(n) = 0$ everywhere apart from the threshold, we can express threshold mixes as well as timed mixes. Thus, such a mix mixes are represented by a single dot in the figure (at $(N, 1)$ for a threshold mix, or $(N, 1 - N_p/N)$ for a pool mix with pool of $N_p$) as it is shown in Figure 1 (d). The mix shown in the figure is a threshold pool mix with threshold $N = 100$ and pool size $N_p = 50$.

Note that the reason we have been able to express all the above mixes in this framework is that they are stateless, i.e. the fraction (and therefore the number) of messages to be flushed depends only on the number of messages in the mix, but not, say, on the number of messages flushed during the previous round.

Before proceeding to examine new $P(N)$ functions, we need to understand the effect they have on the anonymity of a mix.

[1] The current implementation of Mixmaster (version 3) uses a slightly different algorithm: it flushes a fixed fraction of the *total* number of messages in the mix, given that the number of messages that stay in the pool is larger than a minimum; otherwise, it does not send any message.




(a) Timed Mix (b) Timed pool mix

(c) Timed dynamic pool mix (d) Threshold pool mix

**Fig. 1.** Representing mixes as functions from the number of messages inside the mix to the fraction of messages to be flushed

## 3 Anonymity Set Size

We know from [SD02,DSCP02] that the anonymity set size can be computed using the entropy of the probability distribution that relates incoming and outgoing messages. This metric depends on two parameters: the number of messages mixed and the value of the distribution of probabilities of an outgoing message matching an input. In the absence of *a priori* or contextual information about the inputs, this distribution is given by the probability of a message leaving in each round. Therefore, the more messages we mix, the more anonymity; and the more evenly distributed the probability of a message leaving in round $r$, the more anonymity (i.e., we gain anonymity when it is more difficult to predict the number of rounds that a message stays in the pool).

Let us focus on timed pool mixes. The function represented in the Figure 1(b) gives us the probability of a message leaving in the current round as a

function of the number of messages contained in the mix. Let $n_r$ be the number of messages contained in the mix at round $r$, and $P(n_r)$ (the represented function) the probability of a message leaving the mix at round $r$.

The probability of a message that arrived at round $i$ leaving at round $r$ is given by:

$$prob(i) = P(n_r) \prod_{j=i}^{r-1} (1 - P(n_j)) \ .$$

That is, the fact that the message did not leave the mix in the rounds $i..(r-1)$ and it leaves in round $r$. Note that when $P(n_j)$ grows, the $prob(i)$ values are less evenly distributed, and the entropy (and, consequently, the anonymity set size) decreases[2]. This is not a problem if the number of messages mixed at each round is large, but when $n$ is close to the pool size, the anonymity may be too small. We propose a solution to this problem in Section 5.

## 4 Generalising Mixes

The natural way to proceed is to say that a mix is an arbitrary function from the number of messages inside the mix to the percentage of messages to be flushed. What does this gain us?

Throughout the mix literature, a tradeoff between message delay and anonymity can clearly be seen. Indeed, as Serjantov and Danezis showed in [SD02], the pool mix gains more anonymity from higher average delay as compared to the threshold mix. Expressing the mix batching strategy as a function allows us to define an arbitrary tradeoff between anonymity and message delay. We now go on to examine a particular mix function.

## 5 Proposed Design

Suppose that we woudl like to develop a mix which has the properties in low and high traffic conditions[3] as a particular timed dynamic pool mix, but which gains more anonymity for a longer delay in low traffic conditions. This is easily possible – all one needs to do is to invent a suitable function.

Note that the numbers are given on order to illustrate qualitative examples. The values of the functions should be optimised for the requirements of a particular system, depending on the traffic load, number of users, tolerated delay, minimum required anonymity, etc.

In Figure 2 we show a comparison between the timed dynamic pool mix and our new mix, which is defined by a suitable function (normal cumulative distribution function).

[2] Note that this is entirely consistent with our intuition: the higher the fraction of messages we flush each round, the smaller the anonymity. Or equivalently, the more messages we delay during each round, the higher the anonymity.

[3] Or, more pragmatically, the same size of the pool and the same fraction of messages to be sent out when there is lots of traffic.

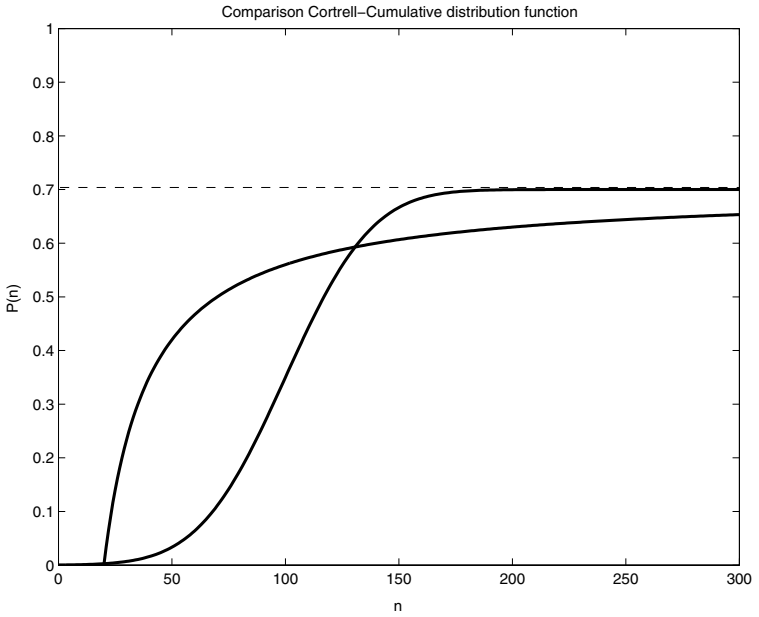



**Fig. 2.** Timed dynamic pool mix vs a mix based on the normal cumulative distribution function

The normal cumulative distribution function has desirable properties. It grows smoothly at low $n$, providing a larger anonymity when the mix contains few messages. This is achieved at the cost of increasing the delay in low traffic conditions. On the other hand, when the number of mixed messages is large enough, the cumulative function improves the delay over the Cottrell function.

## 6 Randomising Mixes: The Binomial Mix

In this section we add randomness to a mix. This has the effect of hiding the number of messages which are in the mix at the time it flushes.

Suppose we treat the result of the function $P(n)$ not as a fraction, but as a probability. We can then use it as the bias of a coin, which we toss for each message inside the mix. A head indicates that this message should be sent out during this round, a tail – that it should remain in the mix.

Let $s$ be the variable that represents the number of messages sent by the mix when it flushes. On average, $s = nP(n)$; but $s$ follows a binomial distribution, which has a variance equal to $np(1-p)$, where $p$ is the result of the function $P(n)$. The property of the mix is that by observing $s$ the attacker does not obtain much information about the value of $n$. The effort required to estimate $n$ is analysed in Section 6.1.

Due to this property, we call this proposed mix *binomial mix*.

### 6.1 Guessing the Number of Messages Contained in the Mix

We analyse the information obtained by a passive attacker that observes the input and output of the binomial mix. Then we explain how the attacker can combine the information obtained in multiple observations and give an estimate of the number of rounds needed to accurately guess $n$.

**Observation of One Output.** When the attacker is allowed to observe only one output, the only available information he has is $s$. We have constructed a simulator that calculates the probabilities of every value of $n$ after observing that the mix outputs $s$ messages.

Given $n$, we can calculate the probability of sending $s$ messages with the following formula, according to the binomial distribution [Fel50]:

$$p(s|n) = \frac{n!}{s!(n-s)!} p^s (1-p)^{n-s} \quad , \tag{1}$$

where $p$ is the result of the function $P(n)$.

But the attacker does not know $n$, he has to estimate $n$ from the observation of $s$. Bayes' rule can be applied to reverse the formula and compute the probability of each $n$[4].

$$p(n|s) = \frac{p(s|n)}{\sum_{i=s}^{N_{max}} p(i|n)} \quad . \tag{2}$$

The attack is implemented as follows: the attacker observes $s$ and assumes that the $n$ that generated this output is at least $s$ and at most $N_{max}$. In order to compute the probability of $n$ taking a particular value, say 100, we apply equation 1 using this value for $n$, and then substitute the result in equation 2. We also need to calculate the result of equation 1 for this $n$ and every possible value of $s$.

Using this formula the attacker can obtain the probability of each value of $n$ given than the mix has sent $s$ messages. The practical results show that the attacker cannot guess the value of $n$ with probability greater than 15%. We have also calculated the 95% confidence interval and found that, typically, it contains between 12 and 30 different values of $n$. This is due to the large value of the variance of a binomial distribution.

**Number of Rounds Needed to Estimate $n$ with 95% Confidence.** We have implemented a passive attack in the simulator in order to have an estimate on the number of rounds required by the attacker to guess with probability 95% the correct value of $n$.

Given that every round is independent from the others, we can multiply the results of every round, taking care of shifting the terms we are multiplying as

[4] Given that the attacker does not have any *a priori* information he must assume, initially, that any possible value of $n$ between $s$ and $N_{max}$ (maximum capacity of the mix) is equally probable.

many positions as the difference between the $n$ of the first round of attack, $n_0$, and the current $n_r$. This difference is known to the attacker because he can count the incoming and outgoing messages. The details of this algorithm can be found in Appendix A.

The attacker, according to the results of the simulations, needs typically close to 200 rounds of observation. This number could be improved by choosing a more appropriate *P(n)-function*. In terms of time, he will have to wait the number of rounds times $T$ (timeout of the mix).

### 6.2 The Blending Attack on the Binomial Mix

As we have seen in the previous section, a passive attacker needs a substantial number of rounds of observation in order to accurately guess the current $n$. Therefore, it does not seem to be practical to deploy a blending attack using the same strategy as with classical pool mixes.

In this section we describe first the attack model, then the steps needed in order to deploy a blending attack and, finally, we analyse the results.

**Attack Model.** The attacker we are considering controls all communication lines (global attacker). He can not only observe all incoming and outgoing messages, but also delay the messages of the users and insert messages (active attacker). The attacker does not have access to the contents of the mix, i.e., the mix is a black box for the attacker (external attacker). In order to test the effectiveness of the design, we consider a setup with only one mix. which

**The Flooding Strategy.** The goal of the attacker is to trace a particular message (the target message) that is sent by a user to the mix. The actions of the attacker can be divided into two phases: the *emptying* phase and the *flushing* phase.

*The Emptying Phase.* During this stage of the attack, the goal of the attacker is to remove all unknown messages contained in the pool, while preventing new unknown messages from going into the mix. In order to force the mix to send out as many unknown messages as possible in each round, the attacker sends to the mix $N_T$ messages, where $N_T$ is the minimum number of messages that guarantees that the $P(n)$ function takes its maximum value, $p_{max}$. If the attacker wants to empty the mix with probability $1 - \epsilon$, then he will have to flood the mix for $r$ rounds.

The formula that can be used to estimate the number of rounds needed to flush all unknown messages with probability $1 - \epsilon$ is:

$$(1 - (1 - p_{max})^r)^n \geq 1 - \epsilon \ . \tag{3}$$

Where $n$ is the number of messages contained in the pool. If the attacker does not have any information about $n$ he will have to assume $n = N_{max}$ (worst case scenario for the attacker).

*Cost of Emptying the Mix.* We compute the cost, $C_E$, of this phase of the attack taking into account the following:

- Number of messages the attacker has to send to the mix.
- Time needed to complete the operation.
- Number of messages the attacker has to delay.

*Number of Messages the Attacker Has to Send to the Mix.* In the first round the attacker has to send $N_T$ messages, to ensure that the function $P(n)$ takes its maximum value, $p_{max}$, and therefore the probability of each message leaving is maximum. In the following rounds, it is enough to send as many messages as the mix outputs. Note that if $n + N_T$ is bigger than $N_{max}$, then some messages will be dropped and the mix will contain $N_{max}$ messages.

Thus, for the first round the attacker sends $N_T$ messages, and the following rounds he sends $(N_T + n)p_{max}$ messages on average. The total number of messages sent during this process is:

$$\text{Number of messages sent} = N_T + (r-1)(N_T + n)p_{max} \ . \tag{4}$$

*Time Needed to Complete the Operation.* This is a timed mix, so the attacker has to wait $T$ units of time for each round. Therefore, the total time needed is $rT$ time units.

*Number of Messages the Attacker Has to Delay.* Assuming that the users generate messages following a Poisson distribution with parameter $\lambda$, the attacker has to delay, in average, $\lambda rT$ messages.

*The Flushing Phase.* Once the mix has been emptied of unknown messages, the attacker sends the target message to the mix. Now, he has to keep on delaying other incoming unknown messages and also send messages to make the mix flush the target.

The number of rounds needed to flush the message is, on average, $r = \frac{1}{p_{max}}$ The cost of this phase is computed according to the previous parameters.

*Number of Messages the Attacker Has to Send to the Mix.* Assuming that the attacker carries out this phase of the attack immediately after the emptying phase, the number of messages needed in the first round is $(N_T + n - 1)p_{max}$, and in the following ones $(N_T + n)p_{max}$. The total number of messages is:

$$p_{max}(N_T + n - 1 + (r-1)(N_T + n)) \tag{5}$$

The other two parameters are computed in the same way as in the emptying phase, taking into account the new value of $r$.

**Guessing the Number of Messages within the Mix with an Active Attack.** The attacker can use the flooding strategy (emptying phase only) in order to determine the number of messages contained in the pool of the mix. This attack is much faster than the one described in Section 6.1, although it requires more effort from the attacker.

**Probabilistic Success.** Note that, due to the probabilistic nature of the binomial mix, the attacker only succeeds with probability $1 - \epsilon$. Therefore, with probability $\epsilon$ there is at least one unknown message in the mix. In this particular case, the attacker can detect his failure if during the flushing phase more than one unknown message leaves the mix in the same round (and there is no dummy traffic policy), which happens with probability $p_{max}^2$ for the case of one unknown message staying during the emptying phase (the most probable case). With probability $p_{max}(1 - p_{max})$ the target message leaves the mix alone, and the attack is successful. Also with probability $p_{max}(1-p_{max})$, the other unknown message leaves the mix first, and the attacker follows a message that is not the target without noticing. Finally, with probability $(1 - p_{max})^2$, both messages stay in the pool and the situation is repeated in the next round.

### 6.3 Average Delay of a Message

Assuming that the population of users generate messages following a Poisson distribution with mean $\lambda$ messages per time unit, and given that the mix flushes messages every $T$ time units, the average number of messages going into the mix per round is $\lambda T$. Assuming that the mix outputs as many messages as it gets (that is, the $P(n)$ function and $N_{max}$ are designed in such a way that the probability of dropping messages because of a lack of space in the mix is very small), the average number of messages sent per round is $s = \lambda T$. We know that $s = nP(n)$, therefore, we have to find $n$ such that $nP(n) = \lambda T$. This number can be found recursively.

Given that the average number of rounds that a message spends in the mix is $\frac{1}{P(n)}$, where $n$ has to be computed as stated above, the average delay of a message going through the binomial mix is $\frac{T}{P(n)}$ time units.

### 6.4 Additional Measure: Timestamps

Additional measures, like timestamps, can be used in order to prevent the blending attack. This idea has already been proposed by Kesdogan *et al.* in [KEB98] for the Stop-and-Go (SG) mixes.

SG mixes work in a different way than pool mixes: users, after choosing the path of mixes, generate a timestamp for each mix in the path that follows an exponential distribution. The message is encrypted several times, each time with the key of one of the mixes. Once an SG mix has received and decrypted a message, it keeps it in the memory a period of time equal to the delay indicated by the user. Then, it forwards the message to the next mix.

**Link Timestamps.** In our design, the user cannot generate timestamps for every mix in the path, because he does not know how long the message is going to be delayed in each mix. Therefore, we propose the use of link timestamps: the user generates a timestamp for the first mix and, in each of the following hops,

the mix puts the timestamp on the message once the message has been taken from the pool and is going to be sent.

When a mix receives a timestamp that is too old, it drops the message. With this policy, the attacker has limited time to delay messages: if he delays the target message too long it will be dropped, and the attacker will not have any means to disclose the recipient of the message.

Using this measure prevents the attacker from delaying the target message at his will, and the attacker does not have means to deploy a blending attack (unless he knows that the message is going to be sent by the user in advance, and can empty the mix before). Therefore, in this scenario the binomial mix provides protection against the blending attack. Furthermore, the anonymity provided by the binomial mix will not be threatened by a change in the traffic load while this change, if large enough, can affect the anonymity provided by a SG mix (since SG mixes only delay messages).

**Drawbacks.** The use of timestamps presents practical problems, and this is the reason why we have not included them in the basic design. The most serious problem is the synchronisation of clocks. If the different computers (both users and mixes) have a deviation in their clocks, many valid messages are dropped. All entities could be synchronised using a time server, but then the security of this time server becomes an issue.

Also, timestamps are not so effective if we are dealing with corrupted mixes: a corrupted mix can put a fake timestamp on a message and give the attacker extra time to empty the following mix in the path.

## 7 Conclusions

We have proposed a framework with which we can generalize classical pool mixes. This model seems to be a powerful tool that gives us a new understanding of the batching strategies implemented by existing mixes. Also, new strategies that improve existing designs arise from the framework. We have proposed a cumulative distribution function in order to have a tailored anonymity/delay tradeoff that adapts to the fluctuations in the traffic load.

We have suggested a simple and intuitive way to deal with the anonymity set size provided by a mix, in which the distribution of probabilities of the number of rounds that a message stays in the pool is a function of $P(n)$.

We have added randomness to the flushing algorithm, in order to hide the number of messages contained in the mix. We have analyzed the effort required by the attacker in order to deploy passive and active attacks. The success of these attacks becomes probabilistic in contrast with classical pool mix designs.

We suggest a timestamp strategy as countermeasure to limit the power of an active attacker. If such an strategy can be securely implemented, the $n-1$ attack becomes no longer possible.

## 8 Future Work

Some of the topics we can identify as deserving further research are:

- The analysis of the possibilities of the framework. We have proposed the cumulative distribution function as an alternative to existing mix algorithms. Other functions with interesting properties may arise from the study of the framework.
- Thorough analysis of the properties of the proposed binomial mix. We have pointed out qualitative properties of this mix. A more in-depth analysis and tests are needed in order to have a full understanding of the design and its possibilities. A method for analysing timed mixes is proposed in [SN03], which needs to be generalised to account for the binomial mix. We would also like to study the implications of the fact that mixes hide the number of messages that are inside the pool.
- Study the properties of the proposed mix when dummy traffic policies are implemented.

## Acknowledgements

Claudia Díaz is funded by a research grant of the K.U.Leuven. This work was also partially supported by the IWT STWW project on Anonymity and Privacy in Electronic Services (APES), and by the Concerted Research Action (GOA) Mefisto-2000/06 of the Flemish Government. Andrei Serjantov acknowledges the support of EPSRC grant GRN24872 Wide Area Programming and EC grant PEPITO.

## References

Cha81. David Chaum. Untraceable electronic mail, return addresses and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

Cot94. L. Cottrell. Mixmaster and remailer attacks, 1994. `http://www.obscura.com/~loki/remailer/remailer-essay.html`.

DSCP02. Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In Paul Syverson and Roger Dingledine, editors, *Privacy Enhancing Technologies*, volume 2482 of *LNCS*, pages 54–68, San Francisco, CA, April 2002. `http://petworkshop.org/2002/program.html`.

Fel50. William Feller. *An introduction to probability theory and its applications.* Wiley, 1950.

Jer00. Anja Jerichow. *Generalisation and Security Improvement of Mix-mediated Anonymous Communication.* PhD thesis, Technischen Universitat Dresden, 2000.

KEB98. D. Kesdogan, J. Egner, and R. Buschkes. Stop-and-go-MIXes providing probabilistic anonymity in an open system. In *Proceedings of the International Information Hiding Workshop*, April 1998.

SD02. Andrei Serjantov and George Danezis. Towards an information theoretic metric for anonymity. In Paul Syverson and Roger Dingledine, editors, *Privacy Enhancing Technologies*, volume 2482 of *LNCS*, pages 41–53, San Francisco, CA, April 2002.
`http://petworkshop.org/2002/program.html`.

SDS02. Andrei Serjantov, Roger Dingledine, and Paul Syverson. From a trickle to a flood: Active attacks on several mix types. In *5th Workshop on Information Hiding*, volume 2578 of *LNCS*, October 2002.

SN03. Andrei Serjantov and Richard E. Newman. On the anonymity of timed pool mixes. In *Workshop on Privacy and Anonymity in Networked and Distributed Systems (18th IFIP International Information Security Conference)*, Athens, Greece, May 2003.

## A Algorithm Used to Combine the Results of Different Observations of the Output

The results of two observations are independent, given that the result of the Bernoulli trials do not depend on previous rounds.

Notation:

- $n_j$ is the number of messages contained in the mix at the $j$-th round of attack (being $n_0$ -the number of messages contained in the mix when the attack starts- the number the attacker is trying to guess).
- $s_j$ is the number of messages sent by the mis in the $j$-th round of attack. This number is a function of $n_i$.
- $f_j$ is the number of messages that arrive to the mix during the $j$-th round. We take into account $f_j$ starting from $j = 1$.
- *shift* is the difference between $n_j$ and $n_0$ ($shift = n_j - n_0$). The attacker knows this number because he observes the number of incoming and outgoing messages at each round; e.g., at round 1 $shift = n_1 - n_0 = f_1 - s_0$. This number can be either positive or negative.
- $P$ is an array that contains the result of the algorithm in the present round, taking into account all the previous rounds. The array has $N_{max}+1$ elements. $P[i]$ contains the probability of $n_0 = i$.
- $A$ is an array that contains the probabilities of the values of $n$ for this round. The array has $N_{max} + 1$ elements. $A[i]$ contains the probability of $n_j = i$, where $j$ is the number of the round.

The algorithm at the $j$-th round is as follows:

$shift > 0$. In this case we know that $n_j > n_0$. In order to be able to multiply the result of this round to the previous ones (which have the maximum value close to $n_0$), we have to shift the values of $A$ *shift* positions to the left. This way, the estimation of $n_j$ can be used to improve our knowledge of $n_0$ ($n_0 = n_j - shift$).

The values we lose at the left of the array are not important, because this corresponds to impossible values of $n_j$: given that $n_0 \geq 0$, this implies that $n_j \geq shift$. On the other hand, at the right side of the array, we have to introduce numbers. The solution is to propagate the value of $N_{max}$. This makes sense

because in case $n_0 \geq N_{max} - shift$ then $n_j = N_{max}$, given that once the capacity of the mix ($N_{max}$) has been exceeded messages are dropped.

After shifting the values of the $A$ array, we have to rescale them in order to have a distribution of probabilities (the sum of all values must be 1).

The code in Java is as follows:

```
if (shift > 0) {
    for (int i=0; i<=N_MAX-shift; i++)
        A[i] = A[i+shift];
    for (int i=N_MAX+1-shift; i<=N_MAX; i++)
        A[i] = A[N_MAX];

    // rescaling A
    double sum = 0.0;
    for (int i=0; i<=N_MAX; i++) sum = sum + A[i];
    for (int i=0; i<=N_MAX; i++) A[i] = A[i]/sum;
}
```

$shift < 0$. This is the case in which in the present round $n_j < n_0$. We have to shift the values of the $A$ array to the right by *shift* positions. We lose the last *shift* values, which are, again, impossible values of $n_j$, because $n_0 \leq N_{max}$ implies $n_j \leq N_{max} - |shift|$. At the left of the array we have to introduce values from the positions 0 to $|shift| - 1$. In this case the value we introduce is 0: we know that $n_j \geq 0$, therefore $n_0 \geq |shift|$ (note that $n_0 = n_j + |shift|$). This implies that any value of $n_0$ smaller than $|shift|$ is impossible.

Again, as in the previous case, we must rescale the values of $A$ in order to obtain the new distribution.

The code in Java is as follows:

```
if (shift < 0) {
    for (int i=N_MAX; i>=-shift; i--)
        A[i] = A[i+shift];
    for (int i=0; i<-shift; i++)
        A[i] = 0.0;
    // rescaling A
    double sum = 0.0;
    for (int i=0; i<=N_MAX; i++) sum = sum + A[i];
    for (int i=0; i<=N_MAX; i++) A[i] = A[i]/sum;
}
```

$shift = 0$. In this case $n_0 = n_j$, and we can multiply both arrays ($P$ and $A$) without changing $A$.

*Multiply P and A.* After shifting and rescaling the elements of the array $A$, we can multiply both arrays element by element. After this multiplication we have to rescale the result and we obtain the distribution of probabilities of the value of $n_0$ including the $j$-th round.

The code in Java is:

```
// multiply probabilities
double sum = 0.0;
for (int i=0; i<=N_MAX; i++) {
    P[i] = P[i]*A[i];
    sum = sum + P[i];
}
// rescaling
for (int i=0; i<=N_MAX; i++) P[i] = P[i]/sum;
```

At this point, the array $P$ contains the current distribution of probabilities, being $P[i]$ the probability of $n_0 = i$, and taking into account the information obtained during all the rounds of attack.

# Modelling Unlinkability

Sandra Steinbrecher[1] and Stefan Köpsell[2]

[1] Freie Universität Berlin, Institut für Informatik,
Takustr. 9, D-14195 Berlin, Germany
steinbrecher@acm.org
[2] Technische Universität Dresden, Fakultät Informatik,
D-01062 Dresden, Germany
sk13@inf.tu-dresden.de

**Abstract.** While there have been made several proposals to define and measure anonymity (e.g., with information theory, formal languages and logics) unlinkability has not been modelled generally and formally. In contrast to anonymity unlinkability is not restricted to persons. In fact the unlinkability of arbitrary items can be measured. In this paper we try to formalise the notion of unlinkability, give a refinement of anonymity definitions based on this formalisation and show the impact of unlinkability on anonymity. We choose information theory as a method to describe unlinkability because it allows an easy probabilistic description. As an illustration for our formalisation we describe its meaning for communication systems.

## 1 Introduction

Every human being sometimes has the desire to act anonymously. Outreach clinics and doctors are visited by many human beings but in some cases visitors do not want others to get to know about their visit. It is quite obvious that someone visiting a doctor or an outreach clinic might do this for a limited number of reasons. For a doctor a patient's reason is linkable to his visitor while for outsiders it should be unlinkable. Obviously a visitor remains anonymous against outsiders regarding a specific visit's reason if his visit or the reason for his visit is unlinkable to him. Sometimes even the visit might indicate the reason, and if only the reason but not the visit is unlinkable to a human being his anonymity is endangered.

Naturally everyone only can be anonymous within a group of human beings that might be in the same situation, especially might have executed a specific action (e.g., visited a doctor).

In contrast to anonymity the notion of (un)linkability is not restricted to human beings and their actions, actions also might be linkable to each other or not. This might endanger a human being's anonymity. One specific action might be unlinkable to a human being but a succession of actions might only be executed by a specific human being and so each single action becomes linkable to the human being. When Clayton et al. studied technical attacks on an electronic

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 32–47, 2003.

student dating service [7] they found out that none of these possible attacks had been executed but some users tried to make 'social' attacks: They asked others for some of their habits or actions. With only a few of these pieces of information linkable to each other it was quite easy to break a user's anonymity.

If a user chooses an unfavourable anonymity group even the links between (some of) his actions might indicate that these actions are (probably) his.

If there is a group of users and a number of actions executed by these users but the concrete links between users and actions are unknown to an attacker the exclusion of users from the group by linking all of his actions to him might reduce the other users' anonymity regarding the remaining actions.

The electronic society that has been built during the recent years gives many human beings a fallacious feeling of acting anonymously. But it becomes even more difficult to act as anonymously as in the society of the real world. Linkable information about a human being that might decrease someone's anonymity can often be collected quite easily in the electronic world.

While someone visiting shops might be anonymous in the real world he might not in the electronic world. For example in the latter he might use non-anonymous payment methods or non-anonymous web surfing. So human beings might become afraid of becoming 'transparent beings' and want to measure (or even better to determine themselves) the degree of anonymity they have in certain situations.

Recently there have been made several attempts to define and formalise the notion of anonymity and unlinkability. Most of the models for anonymity are only formulated for communication scenarios. We give an overview of previous approaches and extend the information theoretic approach to arbitrary scenarios in section 2.

To measure anonymity in real world situations it is necessary to measure the linkability between arbitrary items (e.g., actions, pieces of information, and human beings). The notion of unlinkability and untraceability is well-known in electronic commerce. In section 3 we give an overview of the notion underlying known concepts in electronic commerce and more general scenarios. Based on a general notion for unlinkability we present a formalisation of (un)linkability of arbitrary items and related attacker goals to break unlinkability.

Finally in section 4 is studied which influence these definitions have on a more general measurement of anonymity. In contrast to previous approaches our definitions are not restricted to one specific action but consider a set of actions linkable to a set of actors.

Our formalisation of unlinkability is illustrated by its application on communication systems in several examples. In these examples we assume communication systems to be systems with a set of users who may execute two actions: They may send or receive messages within the system. The users make use of anonymising services (e.g., Anonymizer [1], Crowds [15], Onion-Routing [14], Web mixes [2]) to reach sender and recipient anonymity as well as unlinkability of messages and users. If they do not use such a service users and messages become linkable. We abstract from the internal structure of concrete anonymising

services but concentrate on the formalisation of the unlinkability and anonymity levels they are able to provide. We assume every message to be sent/received by exactly one user. In real world scenarios users might send or receive messages with the same content (for example in the case of web surfing), but we assume these messages to be still technically distinguishable by their internal structure. We further neglect that in real world scenarios one human being might act under the names of multiple users. Every user name involved in the system will be counted as one user and every message sent by a user will be counted as one message.

## 2 Anonymity

A subject only can be anonymous within a group of other subjects. In [12] the following suggestion to standardise the definition of anonymity is given:

> 'Anonymity is the state of being not identifiable within a set of subjects, the anonymity set.'

In real world scenarios a subject's anonymity usually is related to an action. Then the anonymity set is formed by all actors who might have executed the action. The notion further given to measure the anonymity of a subject within such a set is that 'anonymity is the stronger, the larger the respective anonymity set is and the more evenly distributed' the action's execution 'of the subjects within that set is.' I.e., not only the size of the respective anonymity set determines the anonymity of a certain subject but also how likely a subject of the anonymity set might have executed the action.

Usually subjects cannot have the same anonymity against every possible participant and outsider who might be an attacker on the subject's anonymity. Depending on the attacker's knowledge the above set of possible subjects and the likelihood with which they have caused an action varies. For a specific attacker's view anonymity only can decrease. Thus the definition of anonymity in [12] is an analog to the definition of 'perfect secrecy' by Claude E. Shannon [18] as the authors indicate.

There have been made several proposals to describe anonymity with formal languages and logics. Syverson and Stubblebine describe anonymity properties in epistemic language based on group principals [21]. Their description includes the information that should be protected and the nature of the protection (degree of anonymity). This approach is demonstrated with the simple example of an anonymous proxy [1] which removes identifying information of the person requesting a website through it. In [16] a process algebraic formalisation in the modelling language CSP is given. This approach is illustrated with the example of Dining Cryptographers, the DC-Net [4]. Both papers follow the possibilistic approach, that is both metrics only consider the size of the anonymity set not the probability distribution on it.

### 2.1 Anonymity in Communication Systems

For communication systems three types of anonymity can be distinguished [12]: 'Sender/recipient anonymity as the properties that a particular message is not linkable to any sender/recipient and that to a particular sender/recipient, no message is linkable.' Relationship anonymity as the property that it is unlinkable who communicates with whom.

In open environments like the Internet a user is a member of the anonymity set if the probability that he initiated the action is non-zero [11]. But the size of the anonymity set is not sufficient to measure a user's anonymity as already outlined.

Reiter and Rubin [15] introduce a degree of anonymity which they view as 'informal continuum'. Within this continuum they define six degrees that might be reached with 'absolute privacy' as best and 'provably exposed' as worst case. They use these degrees to describe the anonymity their anonymising service Crowds provides. Shmatikov formalises their model by describing Crowds with Markov Chains and expressing the anonymity degrees as temporal probabilistic logic formulas [19].

Based on a mathematical abstraction describing the partial knowledge of a function (the so-called function view [10]) Hughes and Shmatikov develop a modular approach to specify anonymity properties. In this model equivalence relations are used to describe an attacker's inability to distinguish between system configurations (observational equivalence). This approach can be used independent of the underlying algebra or logic. Unfortunately in this approach probabilism is not included.

Information theoretic models can help to precise the above notion of 'the more evenly distributed' by assigning probability distributions to anonymity sets [9,17,8]. These models compare the optimal situation (where every subject in the anonymity set might have executed the action with the same probability) with the situation where the subjects might be assigned different probabilities because of additional information.

In [9,17] only the connection level is considered. In particular they analyse mix-based systems consisting of senders and recipients of messages, mixes used to send/receive these messages anonymously and possible attackers (especially Crowds [15] and Onion-Routing [14] in [9]).

The anonymity on the data level of a communication system is studied in [8]. Their scenario is web surfing of users who are grouped in subsets with different profiles. Every user in a group has the same profile, so when visiting a web site and using this profile this group is his anonymity set. The user remains anonymous but he profits from getting services related to his profile as well as the server profits from placing advertisement fitting the profile.

### 2.2 Anonymity for Arbitrary Actions

Because we want to develop and study a general model of unlinkability between arbitrary items (not restricted to communication systems) we unify and extend the definitions presented in [9,17,8] slightly in this section.

Let $A$ be a non-empty set of actions of arbitrary size and $U = \{u_1, \ldots, u_n\}$ be a set of subjects (the anonymity set regarding a specific action $a \in A$) of size $n$. Every subject $u_i \in U$ with $i \in \{1, \ldots, n\}$ executes $a$ with probability $p_i > 0$.

*Example 1.* In communication systems the set $A$ is defined as $A = \{\text{sending}, \text{receiving}\} \times \{\text{messages}\}_{i \in I}$ with $I$ an index set to enumerate the number of possible messages. In the case of web surfing it simply holds $A = \{\text{requesting}\} \times \{\text{website}_i\}_{i \in I}$. According to [11] in open communication systems like the Internet only subjects whose probability that they have executed the action is non-zero are members of the anonymity set $U$.

Ideally before the execution of action $a$ every $u_i$ will execute it with the a priori probability $\frac{1}{n}$ for a possible attacker's view on the system. This is the basis against which the a posteriori probabilities the attacker assigns to the users is compared in [9,17,8]. The attacker model depends on the concrete application and its requirements. Attackers might get the opportunity to perform several attacks during the execution of the action by which they might get additional information which helps to change the probability distribution on the anonymity set. On the connection level possible attacks are traffic analysis or timing attacks.

For a random variable $X$ let $p_i = P_a(X = u_i)$ denote the attacker's a posteriori probability that given an action $a$, $X$ takes the value $u_i$ (or $u_i$ executed the action $a$). Naturally $\sum_{i=1}^{n} p_i = 1$.

Entropy can be used as a measure to describe the degree of anonymity the system provides against a specific attacker. The attacker's a posteriori entropy is

$$H(X) = -\sum_{i=1}^{n} p_i \log_2(p_i).$$

Serjantov and Danezis [17] define the a posteriori entropy to be the effective size of the anonymity probability distribution $(p_1, \ldots, p_n)$.

Obviously the maximum entropy is

$$\max(H(X)) = \log_2(n).$$

The information the attacker has learned is $(\max(H(X)) - H(X))$. Diaz et al. normalise this information [9] and define

**Definition 1 (Degree of anonymity).** *The degree of anonymity a system provides is*

$$d(U) := 1 - \frac{max(H(X)) - H(X)}{max(H(X))} = \frac{H(X)}{max(H(X))}. \tag{1}$$

The normalisation has the effect that only the probability distribution not the size of the anonymity set is measured in the degree of anonymity. According to definition 1 both an anonymity set $U_i$ with $i > 0$ subjects and a posteriori probabilities $p_1 = \ldots = p_i = \frac{1}{i}$ and an anonymity set $U_j$ with $j > i$ subjects and a posteriori probabilities $q_1 = \ldots = q_i = \frac{1}{j}$ have degree $d(U_i) = d(U_j) = 1$.

The advantage of the normalisation is the the finite range $[0, 1]$ the degree lies within. The degree's maximum/minimum is reached if

$$d(U) = 0 \quad \Leftrightarrow \quad \exists i \in \{1, \ldots, n\} : p_i = 1,$$

$$d(U) = 1 \quad \Leftrightarrow \quad \forall i \in \{1, \ldots, n\} : p_i = \frac{1}{n}.$$

Note both degree and size of an anonymity set have to be given to describe the anonymity a system provides. An alternative is the definition of effective size of an anonymity set [17] above.

## 3 Unlinkability

The notion of anonymity (regarding a specific action) is usually restricted to users, while the notion of unlinkability is applicable to arbitrary items within a given system. For example in communication systems a sender of a message might not be linkable to that message but two messages sent by the same user might be linkable to each other. In [12] the following definition is given:

'Unlinkability of two or more items (e.g., subjects, messages, events, actions,...) means that within this system, these items are no more and no less related than they are related concerning the a priori knowledge.'

Or to say it inversely, items are linkable if they are more or less related than they are without any knowledge of the system. With full knowledge of the system, items within the system are either related to each other or not. Note this description of linkability does describe the inverse of unlinkability but not exactly its usual notion because it includes 'less related'.

An attacker on unlinkability of items within the system a priori knows the items within the system while his knowledge about their relation depends on the concrete system and the concrete attacker. Ideally his view of the system only contains the items.

After the attacker got time to observe/influence the system his knowledge might have increased. A passive attacker only observes the system. Whether he also has the opportunity to become an active attacker and execute several types of attacks influencing the system depends on the concrete system.

The above notion of unlinkability implies that the attacker is successful if his a posteriori probability that items are related has increased or decreased in comparison to his a priori probability. This means not only related items should be protected against detecting this but also unrelated items should a posteriori be related just as much as a priori.

Note the notion of linkability used for electronic payment systems is slightly less restrictive:

'The privacy requirement for the users is that payments made by users should not be linkable (informally, linkability means that the a posteriori probability

of matching is nonneglibly greater than the a priori probability) to withdrawals, even when banks cooperate with all the shops (untraceabiliy). Untraceability guarantees that users remain anonymous, since their identity is only linked to withdrawals.' [3].

Known anonymous cash systems follow this notion.

Digital pseudonyms introduced by Chaum [6] guarantee unlinkability of their use to the corresponding user to make him untraceable. But all transactions executed under the same pseudonym are linkable to each other. If users want to use different pseudonyms for different purposes he should be the only one who is able to link this pseudonyms. The use of credentials [5] enables him to transform statements made about one of his pseudonyms to statements about another one of his pseudonyms while the pseudonyms are still unlinkable to each other for everyone except himself.

In [20] a protocol for unlinkable serial transactions usable in electronic commerce is presented. The tokens (or credentials) used in the protocol fulfill the requirements of users and vendors: both fraud (sharing or abusing tokens) and unlinkability of users' transactions are guaranteed.

In this section we give a formalisation of the above notion of unlinkability. We start with a simple system model for unlinkability within one set in 3.1 and then extend this model to a model for unlinkability between sets which tries to meet real world conditions slightly better in section 3.2. Section 3.3 gives an overview of possible attacker models. If an attacker only learns the numbers of linkable items within a set his a posteriori probabilities of unlinkability will have increased in comparison to his a priori probabilities as we will finally show in section 3.4.

### 3.1 Unlinkability within One Set

Let $A = \{a_1, \ldots, a_n\}$ be the set of items within a given system. For someone with full knowledge of the system some items of this set are related while others are not. We consider a notion of 'is related' that forms an equivalence relation $\sim_{r(A)}$ on the set $A$. Then by this relation $A$ is split in $l$ $(1 \leq l \leq n)$ equivalence classes $A_1, \ldots, A_l$ with $\forall i, j \in \{1, \ldots, l\}$, $i \neq j$: $A_i \cap A_j = \emptyset$ and $A_1 \cup \ldots \cup A_l = A$. Items are related to each other iff they belong to the same equivalence class.

*Example 2 (Communication system).* $A$ could be a set of messages sent. All messages sent by the same sender are related to each other for him but should not for an attacker. But not all relations on $A$ are equivalence relations. Obviously the relation 'sent by the same sender' is one. But the relation 'not sent by the same sender' is no equivalence relation because this relation is neither reflexive nor transitive.

In the following we use this equivalence relation $\sim_{r(A)}$ instead of the notion 'is related' to describe unlinkability of items. An attacker on unlinkability of items within one set knows $A$. A priori he should not know the structure of $\sim_{r(A)}$ but by observing and attacking the system he might learn more about it.

The following example shows the notion 'a priori' here is slightly different to real world scenarios:

*Example 3 (Communication system).* By knowing $A$ a priori the attacker even has an advantage in comparison to real world scenarios where the messages that will be sent usually are not known to an attacker beforehand. But by assuming this knowledge the difference between the ideal situation (the attacker learns nothing) and the imperfect situation (the attacker learns something) can be measured more easily. This assumption is similar to the assumption that the set of possible senders a priori is known to an attacker in open environments. But in real world scenarios he will not have learned the set of senders before they have sent their messages.

**Unlinkability of Two Items within One Set.** For a random variable $X$ let $P(a_i \sim_{r(A)} a_j) := P(X = (a_i \sim_{r(A)} a_j))$ denote the attacker's a posteriori probability that given two items $a_i$ and $a_j$, $X$ takes the value $(a_i \sim_{r(A)} a_j)$ (or $a_i$ and $a_j$ are related). And $P(a_i \not\sim_{r(A)} a_j)$ denotes the analog probability that $a_i$ and $a_j$ are not related. Quite clearly it holds:

$$P(a_i \sim_{r(A)} a_j) + P(a_i \not\sim_{r(A)} a_j) = 1 \quad \forall a_i, a_j \in A. \tag{2}$$

As for the measurement of anonymity we use the attacker's entropy to measure two items' unlinkability. Let $H(i,j) := H(X)$.

**Definition 2 (Degree of unlinkability).** *The degree of $(i,j)$-unlinkability $d(i,j)$ describing the unlinkability of two items $a_i, a_j \in A$ a system provides is*

$$\begin{aligned} d(i,j) &:= H(i,j) \\ &= -P(a_i \sim_{r(A)} a_j) \cdot \log_2(P(a_i \sim_{r(A)} a_j)) \\ &\quad -P(a_i \not\sim_{r(A)} a_j) \cdot \log_2(P(a_i \not\sim_{r(A)} a_j)). \end{aligned}$$

Obviously it holds $0 \leq d(i,j) \leq 1$ and the minimum/maximum is reached if

$$d(i,j) = 0 \quad \Leftrightarrow \quad (P(a_i \sim_{r(A)} a_j) = 1 \quad \vee \quad P(a_i \sim_{r(A)} a_j) = 0)$$

$$d(i,j) = 1 \quad \Leftrightarrow \quad P(a_i \sim_{r(A)} a_j) = P(a_i \not\sim_{r(A)} a_j) = \frac{1}{2}.$$

The latter (the maximum of unlinkability) is the worst case for an attacker: From his view $\forall a_i, a_j \in A$ $(i \neq j)$ the probability that this pair of items is related is equal to the probability it is not.

**Unlinkability of Arbitrary Many Items.** Let $2 < k \leq n$ and $\{a_{i_1}, \ldots, a_{i_k}\}$ be a subset of the set $A$. Then we define $\sim_{r(\{a_{i_1},\ldots,a_{i_k}\})}$ to be an equivalence relation on $\{a_{i_1}, \ldots, a_{i_k}\}$. By this relation $\{a_{i_1}, \ldots, a_{i_k}\}$ is split in equivalence classes. For a random variable $X$ let

$$P\left((\sim_{r(A)} |_{\{a_{i_1},\ldots,a_{i_k}\}}) = (\sim_{r(A)})\right) := P\left(X = \left((\sim_{r(A)} |_{\{a_{i_1},\ldots,a_{i_k}\}}) = (\sim_{r(A)})\right)\right).$$

denote the attacker's a posteriori probability that the equivalence relation $\sim_{r(A)}$ restricted to $\{a_{i_1}, \ldots, a_{i_k}\}$ is the same relation as $\sim_{r(\{a_{i_1},\ldots,a_{i_k}\})}$. This describes the probability that the distribution of the elements $a_{i_1}, \ldots, a_{i_k}$ on equivalence classes in $\{a_{i_1}, \ldots, a_{i_k}\}$ is the same as in $A$. Items that are unlinkable in $\{a_{i_1}, \ldots, a_{i_k}\}$ are unlinkable in $A$ as well.

Figure 1 illustrates this for the small example of $|A| = 11$ with $|A_1| = 3$, $|A_2| = 5$, $|A_3| = 2$, $|A_4| = 1$, the equivalence relation 'having the same filling' and a subset containing 2 items from $A_1$ and 4 items from $A_2$. Items having the same filling in $A$ have so in its subset.

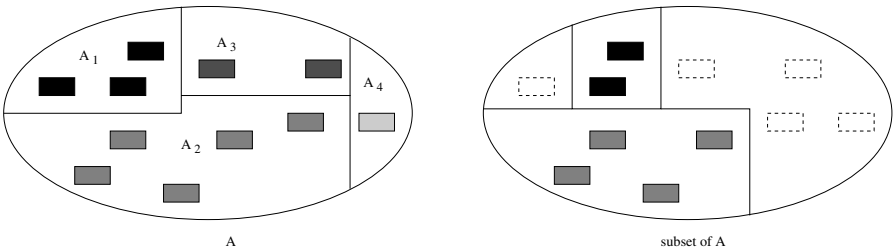



**Fig. 1.** Example: Same distribution on equivalence classes in $A$ and its subset

Let $I_k$ be an index set enumerating all possible equivalence relations on $\{a_{i_1}, \ldots, a_{i_k}\}$. It holds $|I_k| = 2^{k-1}$. The sum over $I_k$ of the above probabilities has to add up to 1 as it does for two items in formula (2):

$$\sum_{j \in I_k} P\left( (\sim_{r_j(A)} |_{\{a_{i_1},\ldots,a_{i_k}\}}) = (\sim_{r(A)}) \right) = 1. \tag{3}$$

Let $H(i_1, \ldots, i_k) := H(X)$. We generalise the degree of unlinkability for two items to arbitrary items:

**Definition 3 (Degree of unlinkability).** *Let $2 < k \leq n$. The degree of $(i_1, \ldots, i_k)$-unlinkability $d(i_1, \ldots, i_k)$ describing the unlinkability of $k$ items $a_{i_1}, \ldots, a_{i_k} \in A$ a system provides is*

$$\begin{aligned} d(i_1, \ldots, i_k) &:= H(i_1, \ldots, i_k) \\ &= -\sum_{j \in I_k} \frac{1}{|I_k|} \Big[ P\left( (\sim_{r_j(A)} |_{\{a_{i_1},\ldots,a_{i_k}\}}) = (\sim_{r(A)}) \right) \\ &\qquad \cdot \log_2 \left( P\left( (\sim_{r_j(A)} |_{\{a_{i_1},\ldots,a_{i_k}\}}) = (\sim_{r(A)}) \right) \right) \Big]. \end{aligned}$$

Obviously it holds $0 \leq d(i_1, \ldots, i_k) \leq 1$ and the minimum/maximum is reached if

$$d(i_1, \ldots, i_k) = 0 \quad \Leftrightarrow \quad \exists j \in I_k : P\left( (\sim_{r_j(A)} |_{\{a_{i_1},\ldots,a_{i_k}\}}) = (\sim_{r(A)}) \right) = 1,$$

$$d(i_1,\ldots,i_k)=1 \quad \Leftrightarrow \quad \forall j \in I_k : P\left((\sim_{r_j(A)} |_{\{a_{i_1},\ldots,a_{i_k}\}}) = (\sim_{r(A)})\right) = \frac{1}{|I_k|}$$

For someone with full knowledge of the system, the set $A$ is split in its equivalence classes uniquely, so is every $\{a_{i_1},\ldots,a_{i_k}\}$. It holds $d(1,\ldots,n)=0$.

### 3.2 Unlinkability between Sets

The example of communication systems shows that the model of unlinkability within one set might not be sufficient to describe communication systems. It does not include the important (un)linkability between specific senders and specific messages. In section 4 we will outline how this model will help to refine anonymity. Now we extend the set of items by the set of all items sending messages, usually the set of senders. Then the set of items consists of items sending messages and being messages. But no item can send and be a message. Example 2 can be refined as follows:

*Example 4 (Communication system).* Let $A = A_s \cup A_m$ with $A_s$ the set of senders and $A_m$ the set of messages within the system. Then the relation 'being sent by the same sender or being this sender' forms an equivalence relation on $A$ with equivalence classes consisting of one item (a sender) of $A_s$ and arbitrary many items of $A_m$.

Whenever $A$ is composed of sets with different types of items it seems to make sense to extend the model of unlinkability within one set to a model for unlinkability between sets.

Let both $A = \{a_1,\ldots,a_k\}$ be a set of items (e.g. actions) and $U = \{u_1,\ldots,u_n\}$ (e.g. users) within a given system. For someone with full knowledge of the system every item in $U$ is related to at least one item in set $A$ and every item in $A$ is related to exactly one item in $U$. It follows that $|A| \geq |U|$.

The notion 'is related' forms a relation $\sim_{r(U,A)}$ between these sets that can be described as an enumeration of pairs.

Concurrently through $\sim_{r(U,A)}$ on $A$ an equivalence relation $\sim_{r(A)}$ is defined as 'is related to the same item in $U$'. This equivalence relation has the same features than the one described in section 3.1. By this relation $A$ is split in $l$ $(1 \leq l \leq n)$ equivalence classes $A_1,\ldots,A_l$ with $A_i \cap A_j = \emptyset \ \forall i,j \in \{1,\ldots,l\}$, $i \neq j$ and $A_1 \cup \ldots \cup A_l = A$. Items are related to each other iff they belong to the same equivalence class.

Figure 2 illustrates this for the small example of $A$ a set of boxes with $|A| = 11$ and $U$ a set of circles with $|U| = 4$. By 'has the same filling' a relation between $A$ and $U$ is defined as well as it defines an equivalence relation on $A$.

In this model Example 4 can be refined as follows:

*Example 5 (Communication system).* Let $U = A_s$ and $A = A_m$. If two items $a \in A$ and $u \in U$ are related to each other by $\sim_{r(U,A)}$ then $u$ sent $a$. If $u \sim_{r(U,A)} a_1$ and $u \sim_{r(U,A)} a_2$ for $a_1, a_2 \in A$ it holds $a_1 \sim_{r(U,A)} a_2$ and these messages are sent by the same sender $u$ and related to each other within $A$ by $\sim_{r(A)}$.

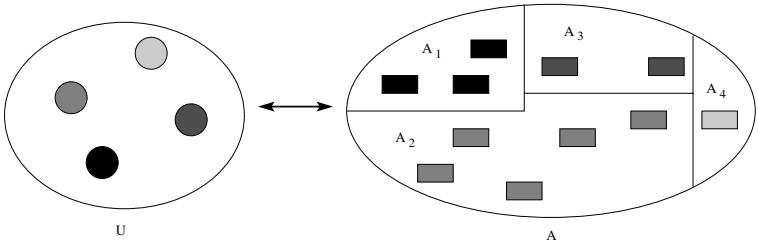



**Fig. 2.** Example

An attacker on unlinkability between sets knows $A$ and $U$. A priori he should not know the structures of $\sim_{r(U,A)}$ and $\sim_{r(A)}$ but by observing and attacking the system he might learn more about it.

**Unlinkability of Two Items.** Accordingly to formula (2) it holds

$$P(u_i \sim_{r(U,A)} a_j) + P(u_i \not\sim_{r(U,A)} a_j) = 1 \quad \forall u_i \in U, a_j \in A.$$

As within one set we define the degree of $(u_i, a_j)$-unlinkability as

$$d(u_i, a_j) = H(u_i, a_j).$$

Optimally $\forall (u_i, a_j) \in U \times A$ the probability that this pair is related to each other is equal to the probability it is not.

The above definitions for unlinkability between two sets can be extended to definitions for unlinkability between arbitrary many sets.

### 3.3 Attacks on Unlinkability

So far we gave only lifeless definitions of unlinkability without exact consideration of attacker goals. These will be given in this section because different attackers might have quite different goals. If an attacker wants to observe a specific victim his attacks might not violate other members of the victim's anonymity set. For example an insurance company trying to find out its (potential) customer's state of health might not collect data about other user's because storing huge amounts of useless data might be expensive.

Security considerations usually distinguish between existential and selective break of a system. We adopt these to the notion of unlinkability introduced:

1. **Existential break:** There exist any two items for which the attacker's a priori probability that they are related to each other is unequal to the a posteriori probability.
2. **Selective break:** The attacker is allowed to choose the items which unlinkability should in- or decrease.

(a) **Chosen Subset of Items:** The attacker may choose a subset of at least two items. For these items his a priori probability that they are related to each other is unequal to the a posteriori probability.
(b) **Chosen Item:** The attacker chooses one item. For this item there exist other items for which the a posteriori probabilities they are related to this specific item are unequal to the corresponding a priori probabilities.

The worst case for unlinkability within one set is that the chosen subset equals $A$ and all a posteriori probabilities either are 0 or 1 in the selective break. For unlinkability between sets accordingly the subset would be $U \cup A$. In authentication or encryption systems existential breaks sometimes are neglected because the attacker success might be no problem for real world applications, e.g., a senseless message with a correct signature does not endanger the system's security. In systems guaranteeing unlinkability linkability between items not selected by the attacker might influence the linkability of items he has selected. In [9,17] several examples are given where anonymity of a specific item is decreased because of this effect. Attackers on unlinkability typically reach their goal by excluding other items to be linkable to the items they are interested in.

### 3.4 The Relation Guaranteeing Unlinkability

While anonymity regarding a specific action depends on the probability distribution on the anonymity set which a priori is uniform, (un)linkability depends on the equivalence classes induced by $\sim_{r(A)}$ or $\sim_{r(A)}$ on the set $A$.

The attacker's knowledge about the structure of the relation $\sim_{r(A)}$ on the given set $A$ of items influence his probability distribution of unlinkability. For instance if the sizes of the equivalence classes are publicly known then optimal a priori probabilities cannot be reached as a posteriori probabilities.

*Example 6 (Communication system).* If an attacker gets to know how many messages every sender sends in the scenario of Example 1 he knows the size of every equivalence class, i.e. $\forall i \in \{1, \ldots, l\}\ |A_i|$ becomes known to the attacker.

The structure of the equivalence classes has an impact on the a posteriori probabilities even in an existential break. The probability that $t$ items $a_{i_1}, \ldots, a_{i_t}$ chosen arbitrarily from $A$ lie in the same equivalence class $A_v$ with $v \in \{1, \ldots, l\}$ is

$$P(a_{i_1} \sim_{r(A)} \ldots \sim_{r(A)} a_{i_t}) = \frac{\sum_{v=1}^{l} \binom{|A_v|}{t}}{\binom{n}{t}}.$$

with $\binom{n}{t} := 0$ for $n < t$.

For the special case $t = 2$ this leads to

$$P(a_{i_1} \sim_{r(A)} a_{i_2}) = \frac{(\sum_{v=1}^{l} |A_v|^2) - n}{n^2 - n}$$

Accordingly $a_{i_1}$ and $a_{i_2}$ lie in different equivalence classes with probability

$$P(a_i \not\sim_{r(A)} a_j) = 1 - \frac{(\sum_{v=1}^{l} |A_v|^2) - n}{n^2 - n}$$

$$= \frac{n^2 - \sum_{v=1}^{l} |A_v|^2}{n^2 - n}$$

**Theorem 1.** *Let $A$ be a set of size $|A| > 1$ and $\sim_{r(A)}$ be an equivalence relation on it. If the sizes of the equivalence classes $A$ is split into is known it cannot be reached that all pairs of items $a_{i_1}$ and $a_{i_2}$ chosen arbitrarily from $A$ have degree of unlinkability $d(i_1, i_2) = 1$.*

**Proof:** For $n \notin \{0, 1\}$ the following requirement holds:

$$d(i_1, i_2) = 1 \Leftrightarrow P(a_{i_1} \sim_{r(A)} a_{i_2}) = P(a_{i_1} \not\sim_{r(A)} a_{i_2})$$

$$\Leftrightarrow \frac{(\sum_{v=1}^{l} |A_v|^2) - n}{n^2 - n} = \frac{n^2 - \sum_{v=1}^{l} |A_v|^2}{n^2 - n}$$

$$\Leftrightarrow 2 \cdot \sum_{v=1}^{l} |A_v|^2 - \left( \sum_{v=1}^{l} |A_v| \right)^2 + \sum_{v=1}^{l} |A_v| = 0$$

$$\Leftrightarrow \sum_{v=1}^{l} \left( |A_v| \left( 2|A_v| - n + 1 \right) \right) = 0.$$

Either all $l$ summands have to equal 0 or the summands have to add up to 0. Because no equivalence class is empty ($\forall i. |A_i| \neq 0$) and $n > 0$ the $v$-th addend of the sum is

1. $> 0$ iff $|A_v| > \frac{n-1}{2}$
2. $= 0$ iff $|A_v| = \frac{n-1}{2}$
3. $< 0$ iff $|A_v| < \frac{n-1}{2}$.

It follows that there exist at most two summands $> 0$:

- If the $v_1$-th and the $v_2$-th summand are $> 0$ it holds $|A_{v_1}| = |A_{v_2}| = \frac{n}{2}$. But it follows $|A_{v_1}| + |A_{v_2}| = |A|$ and the requirement above cannot be fulfilled.
- If only the $v_1$-th summand is $> 0$ it holds $|A_{v_1}| \geq \frac{n}{2}$ and

$$\sum_{v=1, v \neq v_2}^{l} \left( |A_v| \left( 2|A_v| - n + 1 \right) \right) = -|A_{v_1}| \left( 2|A_v| - n + 1 \right) = 0.$$

  And as with $l$ summands it follows either all remaining $l - 1$ summands have to equal 0 or add up to 0. And this can be repeated till $l = 1$ where it will not be fulfilled.
- If no summand is $> 0$ all have to equal 0.
  $\Rightarrow$ All $|A_v|$ have to equal $\frac{n-1}{2}$ and this is impossible.

$\Rightarrow$ There exists no equivalence relation on arbitrary sets $A$ with $|A| > 1$ that guarantees $d(i, j) = 1 \; \forall i, j \in \{1, n\}$.

## 4 Anonymity in Terms of Unlinkability

In terms of unlinkability anonymity in communication systems is defined as 'the properties that a particular message is not related to any sender (recipient) and that to a particular sender (recipient), no message is related.' [12]. The formalisation of this notion of anonymity was given in [9,17] and extended in section 2 to arbitrary actions. This definition is indicated in [12] by anonymity of an item as 'it is not related to any identifier, and the anonymity of an identifier as not being related to any item of interest'.

Please note 'unlinkability is a sufficient condition of anonymity, but it is not a necessary condition' as outlined in [12].

In contrast to previous approaches in real-world scenarios not only the anonymity of an actor within its anonymity set regarding one specific action has to be measured but the unlinkability of a subset of actions and an actor has to be measured for all possible subsets of actions and all actors within a given system. Our definitions from section 3 are suited for this more general scenario.

Recall the definitions for unlinkability between sets from section 3.2. By $\sim_{r(A)}$ the set $A$ is split in $l$ equivalence classes $A_1, \ldots, A_l$. This means every item $u_i$ in $U$ is described uniquely by a subset $A_i \subseteq A$. If (a part of) this unique description becomes known to an attacker and the unlinkability of the items in $A_i$ decreases the item $u_i$'s anonymity decreases.

In communications systems both connection and data level have to be considered to give a measurement of anonymity. The basic approach on the connection level is traffic analysis (overview in [13]). The data level is more difficult to analyze:

*Example 7 (Communication system).* Specific users may have specific interests depending on common personal characteristics e.g., their age, sex, job, religion. These characteristics are often available to the public. And typically additional information is available to an attacker because he usually knows his victim or might influence him [7].

This involves the fact that every item $a_j \in A$ is related to only one item $u_i \in U$. For communication systems this would mean:

*Example 8 (Communication system).* Our definition assumes users not to send exactly the same messages. In mix-based systems [6] this is realistic because this is forbidden to prevent replay attacks. Nevertheless users might send similar contents. But this similarity and the uncertainty about a user's unique description will hopefully prevent an attacker from decreasing the unlinkability of a certain subset to 0 and especially from decreasing a user's anonymity to 0.

Here we come to a point where we only might estimate anonymity and unlinkability because an exact measurement for a single user would assume him knowing how much an attacker knows about his unique description and how much his description varies from other user's description. And to end with an example for this:

*Example 9 (Web surfing).* A user using a unusual combination of operating system and browser and requesting contents not typically for his anonymity group will have a quite low unlinkability degree to his set of web requests. A user should consider this fact when choosing his anonymity group e.g. choosing users of the same age and sex. But because users want to be as anonymous as possible even against members of the same anonymity group this claim might be senseless beneath the fact that the anonymity group might give a single user the fallacious feeling of being anonymous (flooding attack or a social variant of it).

## 5 Conclusion

We generalised the definitions for anonymity [9,17,8] to arbitrary scenarios, and we gave new definitions for unlinkability based on the notions in [12]. Especially we have shown there exists no equivalence relation on trivial sets that guarantee the best possible unlinkability in an existential break if only the sizes of the equivalence classes have become known to an attacker. Our next task will be to study sub-optimal equivalence classes on given sets. Finally we refined anonymity in terms of unlinkability. Especially we pointed out the limits of measuring anonymity in real world applications. In future work we will try to evaluate the impact of different constructions of users' unique descriptions on the connection and data level on their anonymity within the system.

## Acknowledgements

We would like to thank Andrei Serjantov, Andreas Pfitzmann and the anonymous reviewers for valuable suggestions and their patience.

## References

1. The anonymizer. http://www.anonymizer.com.
2. Oliver Berthold, Hannes Federrath, and Stefan Köpsell. Web mixes: A system for anonymous and unobservable internet access. Designing Privacy Enhancing Technologies. Proc. Workshop on Design Issues in Anonymity and Unobservability, LNCS 2009, Springer-Verlag, Heidelberg 2001, pp. 115–129.
3. Stefan Brands. An efficient off-line electronic cash system based on the representation problem. Centrum voor Wiskunde en Informatica, Computer Science/Department of Algorithmics and Architecture, Report CS-R9323, March 1993.
4. David Chaum. The dining cryptographers problem: unconditional sender and recipient untraceability. Journal of Cryptology (1), 1988.
5. David Chaum. Showing credentials without identification - signatures transferred between unconditionally unlinkable pseudonyms. Advances in Cryptology - EUROCRYPT 85, LNCS 219, Springer-Verlag Berlin 1986, pp. 241–244.
6. David Chaum. Untraceable electronic mail, return addresses and digital pseudonyms. Communications of the ACM, 24(2), 1981, pp. 84–88.

7. Richard Clayton, George Danezis, and Markus G. Kuhn. Real world patterns of failure in anonymity systems. Information Hiding 2001, LNCS 2137, Springer-Verlag Berlin 2001, pp. 230–245.
8. Claudia Diaz, Joris Claessens, Stefan Seys, and Bart Preneel. Information theory and anonymity. Proceedings of the 23rd Symposium on Information Theory in the Benelux, May 29–31, 2002, Louvain la Neuve, Belgium.
9. Claudia Diaz, Stefan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. Privacy Enhancing Technologies 2002, LNCS 2482, Springer-Verlag Berlin.
10. D. Hughes and V. Shmatikov. Information hiding, anonymity and privacy: A modular approach. To appear in Journal of Computer Security, 2003.
11. Dogan Kesdogan, Jan Egner, and Roland Büschkes. Stop-and-go-mixes providing probabilistic anonymity in an open system. Information Hiding 1998, LNCS 1525, Springer-Verlag Berlin 1998, pp. 83–98.
12. Marit Köhntopp and Andreas Pfitzmann. Anonymity, unobservability, and pseudonymity - a proposal for terminology. Draft v0.12., June 2001.
13. Jean-Francois Raymond. Traffic analysis: Protocols, attacks, design issues, and open problems. Privacy Enhancing Technologies 2000, LNCS 2009, Springer-Verlag Berlin.
14. M.G. Reed, P.F. Syverson, and D. Goldschlag. Anonymous connections and onion routing. IEEE Journal on Selected Areas in Communication, Special Issue on Copyright and Privacy Protection, 1998.
15. M. K. Reiter and A. D. Rubin. Crowds: Anonymity for web transactions. ACM Transactions on Information and System Security 1(1), November 1998, pp. 66–92.
16. Steve Schneider and Abraham Sidiropoulos. CSP and anonymity. ESORICS 1996, LNCS 1146, Springer-Verlag Berlin 1996, pp. 198–218.
17. Andrei Serjantov and George Danezis. Towards an information-theoretic metric for anonymity. Privacy Enhancing Technologies 2002, LNCS 2482, Springer-Verlag Berlin.
18. C. E. Shannon. ommunication theory of secrecy systems. The Bell System Technical Journal 28/4 (1949), pp. 656–715.
19. Vitaly Shmatikov. Probabilistic analysis of anonymity. Proc. 15th IEEE Computer Security Foundations Workshop (CSFW) 2002, pp 119–128.
20. Stuart G. Stubblebine, Paul F. Syverson, and David M. Goldschlag. Unlinkable serial transactions: Protocols and applications. ACM Transactions on Information and System Security, Vol. 2, No. 4, Nov.1999, pp. 354–389.
21. Paul F. Syverson and Stuart G. Stubblebine. Group principals and the formalization of anonymity. FM'99 – Formal Methods, Vol. I, LNCS 1708,, Springer-Verlag 1999pp. 814–833.

# Metrics for Traffic Analysis Prevention

Richard E. Newman[1], Ira S. Moskowitz[2],
Paul Syverson[2], and Andrei Serjantov[3]

[1] CISE Department, University of Floridam
Gainesville, FL 32611-6120, USA
nemo@cise.ufl.edu
[2] Center for High Assurance Computer Systems, Code 5540
Naval Research Laboratory
Washington, DC 20375, USA
{moskowitz,syverson}@itd.nrl.navy.mil
[3] University of Cambridge Computer Laboratory
Cambridge CB3 0FD, United Kingdom
Andrei.Serjantov@cl.cam.ac.uk

**Abstract.** This paper considers systems for Traffic Analysis Prevention (TAP) in a theoretical model. It considers TAP based on padding and rerouting of messages and describes the effects each has on the difference between the actual and the observed traffic matrix (TM). The paper introduces an entropy-based approach to the amount of uncertainty a global passive adversary has in determining the actual TM, or alternatively, the probability that the actual TM has a property of interest. Unlike previous work, the focus is on determining the overall amount of anonymity a TAP system can provide, or the amount it can provide for a given cost in padding and rerouting, rather than on the amount of protection afforded particular communications.

## 1 Introduction

Previous attempts to gauge anonymity provided by an anonymous communication system have been focused on the extent to which the actions of some entity are protected by that system. For example, how well protected is the anonymity of the sender of an arbitrary message, or its recipient, or the connection of sender and recipient, etc. [11,18]. Various ways to measure such protection have been proposed from the classic *anonymity set* to cryptographic techniques [12], probabilistic measures [14], and information theoretic measures [3,15].

The focus of this work is a bit different from all of those. Rather than examine how well protected the actions of a particular agent (or pair of agents) are, we will examine how much protection a system provides to all its users collectively. Put too succinctly, previous work has focused on how well the system distributes available anonymity, while we focus on the amount of anonymity there is to distribute.

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 48–65, 2003.

We consider a system of $N$ nodes wanting to send (a large number of) end to end encrypted messages to one another over an underlying network[1]. These $N$ sender nodes cooperate to try to prevent the adversary from performing traffic analysis by using padding and rerouting. While fielded Traffic Analysis Prevention (TAP) systems are likely to be limited in their ability to so cooperate, padding and rerouting are commonly proposed means to counter traffic analysis [1,2,13,19]. Yet, there has been no theoretical analysis of how much protection is possible using padding and rerouting techniques. Our model allows assessment of upper bounds on what any system can accomplish by such means.

Our central means to examine anonymous communication is the *traffic matrix* (TM), which represents all end-to-end message flows. One can examine the difference between observed traffic matrices and the traffic matrix of an ideal system to determine how much an adversary might gain from observing the system. Alternatively, the difference between observations on a protected system and an unprotected system can be examined to determine the amount of protection afforded. Traffic matrices allow us to measure the communication costs of TAP methods, which gives us a potential means of comparing the costs and benefits of various TAP methods and systems.

This paper uses an information-theoretic, entropy-based approach to measuring the success of a TAP system, much as Shannon used entropy to measure the success of a cryptosystem [16]. The goal of the group of nodes sending messages to one another is to make the number of possible traffic matrices (TMs) large enough and the probability that the actual TM is determined from what is observed low enough that the observations are essentially useless to the adversary. If the adversary has no *a priori* means of excluding any particular TM (which may depend on the measurement interval and the expectations of traffic), then the possible TMs are not just all TMs that are dominated by the observed TM, but all that have a rerouted TM that is dominated by the observed TM. These terms will be made precise in subsection 2.2.

Previous methods of TAP have either used rerouting or padding or both (in addition to padding messages to a constant length and payload encryption) to achieve TAP. In general, the effects of these controls are to

a. increase the total amount of traffic;
b. increase the cryptographic processing load on the involved nodes;
c. mask the true source and destination of individual messages;
d. make the number of possible true traffic patterns very large.

While traditional link encryption and padding to the link speed at the link level is perfect at concealing the true traffic patterns, it has many deficiencies. It requires that all routers in the network participate and remain secure, and that all are willing to saturate their links with apparent traffic, whether or not there is actual traffic to send. The more efficient "Neutral TM" approach used by Newman-Wolfe and Venkatraman [8,21] still increases traffic to around twice its original level, depending on the spatial traffic distribution [9,20]. Onion routing [10,5,19] increases traffic greatly as well, by routing a packet through several

[1] The network graph is not necessarily complete.

(usually at least five) onion routers. One might expect this to increase the aggregate traffic by the number of onion routers the packet traverses (i.e., make the total load five times higher in this case)[2].

This paper considers the information that is available in the static, spatial traffic information to a global passive adversary when transport level padding and rerouting are employed.

## 2 Adversary Model

As in much previous work, we assume a global passive adversary who can observe all traffic on all links between all nodes, that is all senders, receivers, and any intermediate relay points the system may contain.

Since she observes all message flows, the global passive adversary is very strong, perhaps stronger than any likely real adversary. On the other hand she mounts no active attacks, which makes her weaker than many likely real adversaries. However, our concern is to first describe means to determine a bound on anonymity capacity of a system even if that bound is not likely to be reached in practice.

Since we are only addressing TAP, we assume no one can track redirected messages through an intermediate node by recognizing its format or appearance. Similarly, no one is able to distinguish padding messages from 'genuine' traffic. Of course, a node that is a redirection intermediary knows which incoming message correlates with which outgoing message, and nodes that generate and/or eliminate padding can recognize it locally.

Our adversary is thus best thought of as having a traffic counter on all the wires between nodes. The units of traffic may be generically described as *messages*. If necessary, traffic may also be measured in bits. The rate at which these counters are checked governs the granularity of the picture of traffic flows that the adversary has. The degree of synchronization on those link clocks (i.e., whatever governs the frequency at which each link is checked), will also determine the granularity of the causal picture that the adversary has. For example, an adversary may be able to recognize or dismiss possible message redirections by observing the relative timing of flows into and out of a node. However, for the purposes of these initial investigations, we will consider the period of observation to be sufficient for all actual traffic, as well as dummy messages and rerouted actual traffic, to be delivered and counted.

Note that there is some degree of noise or uncertainty due to the nature of measurement of traffic — it is not instantaneous but must be measured over some period of observation (window). Both the size of the window and the window alignment will affect the measurements and their variation. This argues for decreased resolution in the measured values (e.g., the difference between 68,273 packets and 67,542 packets may be considered to be below the noise threshold in the measured system; likewise, byte count numbers may also only be of use up to two or three digits). Study of the levels of "noise" in the measured system

[2] The actual load increase depends on the underlying network and the routes taken.

and "noise" in the measurement methods is needed to make a valid estimate of the appropriate level of resolution for the measurements. This paper assumes such considerations out of the model.

### 2.1 Network and Adversary Assumptions

For purposes of this paper, we make a number of assumptions.

- All nodes may send, receive, or forward traffic. Thus, we do not differentiate between senders, receivers, and virtual network elements. This is most typically true of a peer-to-peer system; however, this could also reflect communication within an anonomizing network where the outside connections are either invisible or ignored.
- All links (directed edges) have a constant fixed-bound capacity (in messages that can be sent in some unit of time). The number of messages that can be passed over any (simplex) network link is the same. Any padding or redirection a node passes over a link will reduce the number of messages it can initiate over that link.
- All link traffic counters are checked once (simultaneously).

This last assumption means that we do not capture any timing information or causal connections between message flows. Even with this simplifying assumption there is more than enough complexity in the network traffic information for an initial investigation. Further, as we have noted, a primary purpose of this work is to set out means to describe the anonymity capacity of a network. This assumption allows us to consider the temporally coarsest adversary of our model. Any temporal information that a finer adversary could use will only serve to lower such a bound. While such a coarse-grained adversary is inherently interesting and may even be realistic for some settings, obviously the study of an adversary that can take advantage of timing information is ultimately important. Such refinement of assumptions is possible within our general model, and we leave such questions for future work.

### 2.2 Definitions

Now we define some terms.

**Traffic Matrix (TM).** An $N \times N$ non-negative integer matrix $T$ in which cell $T[i, j]$ holds the number of messages sent from node $i$ to node $j$ in the period of observation. The diagonal entries are all zero.

**Domination.** One traffic matrix $T$ dominates another traffic matrix $T'$ iff $\forall i, j \in [1..N], T[i, j] \geq T'[i, j]$.

**Neutral TM.** A traffic matrix in which all of the non-diagonal values are equal. The unit neutral TM is the neutral TM in which all the non-diagonal values are ones. The magnitude of a neutral TM is the constant by which the unit TM must be multiplied to equal the neutral TM of interest.

**Actual TM, $T_{act}$.** The end-to-end traffic matrix, neither including dummy messages nor apparent traffic arising from rerouting through intermediate nodes; the true amount of information required to flow among the principals in the period of observation.

**Observed TM, $T_{obs}$.** The traffic matrix that results from treating all and only observed flows on links as reflecting genuine traffic, i.e., all padding is treated as genuine traffic and redirection is treated as multiple genuine one hop messages.

**Routes, Flow Assignments.** If the actual traffic matrix specifies that $T[i, j]$ messages must be sent from node $i$ to node $j$ in a period of time, then these messages must be routed from node $i$ to node $j$ either directly or indirectly. A route from node $i$ to node $j$ is a path in the network topology graph starting at node $i$ and ending at node $j$. A flow assignment specifies for each path used to send messages from node $i$ to node $j$ how many of the messages are delivered using that path.

**Link Load.** The load on a (simplex) link is the sum of the number of messages delivered by the flow assignments over paths that include that link. For a flow assignment to be feasible, the load on a link must not exceed its capacity.

**Total Traffic Load.** Total traffic load in an $N \times N$ traffic matrix $T$ is

$$L(T) = \sum_{i,j \in [1..N]} T[i, j].$$

where $[1..N]$ is the set of integers between 1 and $N$, inclusive. That is, the total (or aggregate) load is just the sum of the link loads.

**Feasible TM.** These TMs are the only ones for which there are corresponding routes with flow assignments for which the combined flows on a given link in the graph do not exceed its capacity.

## 3 Observations

First, we notice that, depending upon $T_{obs}$, there are limits to what the true traffic matrix can be, no matter what the TAP techniques might be used. For example, if a node $A$ in $T_{obs}$ has a total incoming flow of $f_{in,T_{obs}}(A)$,

$$f_{in,T_{obs}}(A) \triangleq \sum_{i=1}^{N} T_{obs}[i, A],$$

then the total incoming flow for the same node $A$ in $T_{act}$ is bounded by that same total, that is,

$$f_{in,T_{act}}(A) \leq f_{in,T_{obs}}(A).$$

This is true because the observed incoming flow includes all of the traffic destined for $A$, as well as any dummy packets or redirected messages for which $A$ is the intermediate node. For similar reasons, the outgoing flow of any node $A$ in $T_{act}$ is bounded by the observed outgoing flow in $A$.

The topology (graph connectivity) of the network and the link capacities limit the possible traffic matrices that can be realized. As noted, feasible TMs are the only ones for which there are corresponding routes with flow assignments for which the combined flows on a given link in the graph do not exceed its capacity. Based on the limitations of the network, the set of possible traffic matrices is therefore finite (if we consider integer number of packets sent over a period of observation). Define the set of possible traffic matrices for a network represented by a directed graph $G =< V, E >$ with positive integer edge[3] weights $w : E \to \mathbb{N}$ to be

$$\mathbb{T}_{<G,w>} = \{T \mid T \text{ is feasible in } < G, w >\}$$

The graphs we consider are cliques, but a node $A$ may be able to send more data to node $B$ than the link directly from $A$ to $B$ can carry, by sending some of the messages through an intermediate node.

Beyond the limits of the network itself, our adversary is able to observe all of the traffic on the links, and from observations over some period of time, form an observed traffic matrix, $T_{obs}$. As previously noted, since any traffic matrix $T$ reflects the end-to-end traffic between nodes, $T_{obs}$ can be thought of as reflecting the pretense that there are no messages sent indirectly, i.e., all messages arrive in one hop. The observed traffic matrix further limits the set of actual traffic matrices possible, as they must be able to produce the observed traffic matrix after modifications performed by the TAP system. For example, it is not feasible for the total traffic in the actual TM to exceed the total traffic in the observed TM.

Let the set of traffic matrices compatible with an observed TM, $T_{obs}$ be defined as

$$\mathbb{T}_{T_{obs}} \triangleq \{T \mid T \text{ could produce } T_{obs} \text{ by TAP methods}\}$$

Note that $\mathbb{T}_{T_{obs}} \subseteq \mathbb{T}_{<G,w>}$, since the observed traffic matrix must be feasible, and that $T_{act}, T_{obs} \in \mathbb{T}_{T_{obs}}$.

We now describe the affect of TAP methods in determining $\mathbb{T}_{T_{obs}}$. Further details on the TAP transforms themselves are presented in section 6. A *unit padding transform* reflects adding a single padding message on a single link and results in incrementing, by one, the value of exactly one cell of a traffic matrix. A *unit rerouting transform* reflects redirecting a single message via a single other node. So, rerouting one unit of traffic from $A$ to $B$ via $C$ causes the traffic from $A$ to $B$ to decrease by one unit, and the traffic from $A$ to $C$ and from $C$ to $B$ each to increase by one unit. This causes the traffic in the new TM to remain constant for $A$'s row and for $B$'s column, but to increase by one unit for $C$'s column and $C$'s row ($C$ now receives and sends one more unit of traffic than before). The total load therefore increases by one unit also (two unit increases

[3] Edge weights can be considered the number of packets or the number of bytes that a link can transfer over the period of observations. We can also consider node capacities, which could represent the packet switching capacity of each node, but for now consider this to be infinite and therefore not a limitation.

and one unit decrease for a net of one unit increase — we replaced one message with two).

We say that a traffic matrix $T$ is $P$-derivable from traffic matrix $T'$ iff $T$ is the result of zero or more unit padding transforms on $T'$. We say that a traffic matrix $T$ is $k-P$-derivable from traffic matrix $T'$ iff $T$ is the result of exactly $k$ unit padding transforms on $T'$. This is true iff $\forall i, j\ T'[i,j] \leq T[i,j]$ and

$$L(T) = L(T') + k$$

Note that the set of $P$-derivable traffic matrices from some TM $T$ is the union for $k = 0$ to $L(T)$ of the sets of $k-P$-derivable TMs relative to $T$.

We say that a traffic matrix $T$ is $R$-derivable from another traffic matrix $T'$ iff $T$ is the result of zero or more unit rerouting transforms on $T'$. We say that a traffic matrix $T$ is $k-R$-derivable from another traffic matrix $T'$ iff $T$ is the result of exactly $k$ unit rerouting transforms on $T'$. The set of $R$-derivable traffic matrices from some TM $T$ is the union for $k = 0$ to $L(T)$ of the sets of $k-R$-derivable TMs relative to $T$.

We say that a traffic matrix $T$ is $R, P$-derivable from another traffic matrix $T'$ iff $T$ is the result of zero or more unit padding or rerouting transforms on $T'$. We say that a traffic matrix $T$ is $k-R, P$-derivable from another traffic matrix $T'$ iff $T$ is the result of exactly $k$ unit padding or rerouting transforms on $T'$. The set of $R, P$-derivable traffic matrices from some TM $T$ is the union for $k = 0$ to $L(T)$ of the sets of $k-R, P$-derivable TMs relative to $T$.

In general, padding and rerouting transformations may be described as addition of specific unit transformation matrices to a given TM. This will be explored further in section 6. Note that, in most cases, padding and rerouting operations commute[4].

## 4 Problem Statement

This section defines the problems considered. In this model, the "sender" consists of all of the $N$ nodes listed in the traffic matrix, which cooperate to try to disguise an actual traffic matrix $T_{act}$ by performing TAP operations to produce the traffic matrix $T_{obs}$ observed by the global, passive adversary. This aggregate sender must deliver all of the messages required by $T_{act}$ in the period of observation, and we assume there is sufficient time to do this.

### 4.1 Sender

The aggregate sender is given the actual TM, $T_{act}$, and must produce the set of TAP transformations on it to create the observed TM, $T_{obs}$. The sender may be under some cost constraints (in which case the goal is to create the greatest amount of uncertainty in the adversary possible within the given budget), or may be required to create an observed TM, $T_{obs}$, that meets some goal of obfuscation (at a minimum cost).

[4] If a padding message may then be rerouted, then padding first offers more options for the subsequent rerouting. We do not consider this useful, and limit rerouting to actual traffic.

### 4.2 Adversary

The adversary may ask generically the following question, "Is $T_{act} \in \mathbb{T}^*$?," where $\mathbb{T}^* \subseteq \mathbb{T}_{<G,w>}$ is some set of TMs of interest to the adversary. Note that $\mathbb{T}^*$ may be a singleton, which means that the adversary has some particular TM in which he has interest, and through a series of such questions, the adversary can attempt to determine the actual TM, $T_{act}$, exactly. More often, the adversary may not care about some of the communicating pairs, and may not even care about the detailed transmission rates between the pairs of interest.

In general, the property $\mathbb{T}^*$ can be given as the union of sets of the form

$$\mathbb{T}^*_k = \{T | \alpha_{i,j,k} \leq T[i,j] \leq \beta_{i,j,k} \ \forall i,j = 1,2,...,N\} \ ,$$

i.e., a range set, in which the values of the cells of the TM are constrained to lie within some range. So

$$\mathbb{T}^* = \bigcup_k \mathbb{T}^*_k \ .$$

Observe that the set of these range sets is closed under set intersection, that is, the intersection of two range sets results in another range set[5].

It may be more apropos to rephrase the question as, "What is the probability that the actual TM has the property of interest, given the observed TM," i.e., $Pr(T_{act} \in \mathbb{T}^* \mid T_{obs})$, since under most circumstances, whether or not $T_{act}$ is in $\mathbb{T}^*$ cannot be known with certainty.

$$Pr(T_{act} \in \mathbb{T}^* \mid T_{obs}) \ = \ \sum_{T \in \mathbb{T}^*} Pr(T|T_{obs}) \ .$$

Absent *a priori* information to give one possible TM (i.e., one consistent with the observations), a greater likelihood of having been the actual TM, we can give all those TMs consistent with the observed TM equal weight, so that

$$Pr(T|T_{obs}) \ = \ \frac{1}{|\mathbb{T}_{T_{obs}}|} \ .$$

This is the maximum entropy result, with

$$Pr(T_{act} \in \mathbb{T}^* \mid T_{obs}) \ = \ \frac{|\mathbb{T}_{T_{obs}} \cap \mathbb{T}^*|}{|\mathbb{T}_{T_{obs}}|} \ .$$

Adversary possession of *a priori* information may reduce anonymity in two ways.

1. She may limit $\mathbb{T}_{T_{obs}}$ further by using knowledge about this instance of $T_{act}$, [6] e.g., "At least one of the nodes did not send any real traffic." Such constraints on $\mathbb{T}_{T_{obs}}$ may be expressed by using the same techniques as we used to express matrices of interest, $\mathbb{T}^*$.

[5] These kinds of properties may be of interest to adversaries exercising a network covert channel.

[6] We can then estimate the amount of information that the observations give to the adversary in terms of the relative entropy from the knowledge to the observations.

2. She may alter relative probabilities of the TMs within $\mathbb{T}_{T_{obs}}$ (which leads to submaximal entropy). Examples of this include the adversary possessing a probability distribution over the total amount of traffic in $T_{act}$ or the total cost which the sender is prepared to to incur to disguise the actual traffic matrices (see Section 5.2). Indeed, the adversary may even possess a probability distribution over the $T_{act}$ that she expects will occur.

So, in the end, it is not necessary to make the observed traffic matrix, $T_{obs}$, neutral; it is enough to disguise $T_{act}$ so that the adversary's knowledge of its properties of interest are sufficiently uncertain.

## 5 Traffic Analysis Prevention Metrics

This section considers the degree to which the sender can make the adversary uncertain regarding the nature of $T_{act}$. First, it considers the costs of performing TAP operations, then considers the strategies the sender may have, and the effects of these on the adversary's knowledge. Finally, the effects of *a priori* knowledge by the adversary are evaluated.

### 5.1 Cost Metrics

Rerouting and padding are not free operations. Unit padding adds one more message from some source to some destination in the period (increasing exactly that cell by one unit and no others). Unit rerouting from node $A$ to node $B$ via node $C$ decreases the traffic from $A$ to $B$ by one unit, but increases the traffic from $A$ to $C$ and from $C$ to $B$, without changing any other cells. Hence in both cases, in this model, they increase the total load by one unit of traffic.

The simplest cost metric for disguising traffic is just the change in the total traffic load from the actual to the observed TM. Let $T_1$ and $T_2$ be two traffic matrices, and define the distance between them to be

$$d(T_1, T_2) = |L(T_1) - L(T_2)|.$$

In the simplest case, the cost is just the distance as defined above. In general, the cost may be non-linear in the distance, and may be different for padding than for rerouting[7]. For the remainder of this paper, we will only consider the simple case.

[7] Padding and rerouting costs may not be the same if node computation is considered. It may be much easier for a node that receives a dummy message to decode the encrypted header and determine that the remainder of the message is to be discarded than it is for the node to decrypt and reencrypt the message body, create an appropriate TAP header and network header, then form the forwarded message and send it on the the true destination.

### 5.2 Sender Strategies

Making changes to the actual traffic matrix by rerouting and padding will increase the total traffic load in the system, and the sender may not wish to incur large costs. Sender strategies may be thought of in two factors. The first factor is whether a neutral traffic matrix is sent every period, or whether a non-neutral observed traffic matrix is acceptable. The second factor is whether or not the sender adapts the costs it is willing to incur to the actual traffic it must send. These are not unrelated, as is explained below.

If the observed traffic matrix is always made neutral, then the sender must use a total load sufficient to handle the peak amount of traffic expected (modulo traffic shaping[8]), and must alway reroute and pad to that level. Often, the total traffic load of the observed traffic matrix will be many times larger than the total traffic load of the actual traffic matrix, and the sender will just have to live with these costs. The advantage of this is that the adversary never learns anything; the traffic always appears to be uniform and the rates never vary.

If the set of actual TMs to be sent is known to the sender in advance, then an adaptive strategy may be used to minimize the total cost. The "peaks" in the actual TMs are flattened using rerouting. Then the maximum matrix cell value over all of the TMs resulting from rerouting is chosen as the amplitude of the neutral TMs to send for that sequence.

Mechanisms for dynamically handling changing load requirements are considered in Venkatraman and Newman-Wolfe [21]. Here, the sender may change the uniform level in the neutral traffic matrix, adjusting it higher when there are more data to send and lower when there are fewer. This will reduce the costs for disguising the actual traffic patterns. However, the sender should avoid making frequent adjustments of small granularity in order to avoid providing the adversary with too much information about the total actual load[9].

If non-neutral traffic matrices are acceptable, the sender can either set a cost target and try to maximize the adversary's uncertainty, or can set an uncertainty target and try to minimize the cost of reaching it. Regardless, the goal is to keep the amortized cost of sufficiently disguising the actual TMs reasonable. In the former case, a non-adaptive strategy can be employed, in the sense that the cost will not depend on the actual traffic matrix. If the sender always uses the same cost for each period, and the adversary knows this cost, then this severely reduces the entropy for the adversary. Here, the adversary need only consider the intersection of a hypersphere and $\mathbb{T}_{T_{obs}}$. That is, the adversary knows that

$$T_{act} \in \{T \in \mathbb{T}_{T_{obs}} | d(T, T_{obs}) = c\},$$

where $c$ is the cost (known to the adversary) that the sender incurs each period.

[8] In traditional networking, traffic shaping is a form of flow control that is intended to reduce the burstiness and unpredictability of the traffic that the sources inject into the network so as to increase efficiency and QOS [6,4,17]. In TAP networks it is used to hide traffic flow information [1].

[9] A "Pump"-type [7] approach may be taken to lessen the leaked information.

A better non-adaptive strategy is to pick a distribution for the costs for each period, then generate random costs from that distribution. Once a cost is picked, then the entropy associated with the observed TM (with respect to the properties of interest, if these are known by the sender) can be maximized. The adversary then has to consider the intersection of a ball with $\mathbb{T}_{T_{obs}}$ rather than a hypersphere. In this fashion, the mean cost per period can be estimated, and yet the adversary has greater uncertainty about the possible actual TMs that lead to the observations.

When the total traffic is very low, the sender may be willing to incur a greater cost to pad the traffic to an acceptably high level, and when the actual TM already has a high entropy (for the adversary), then it may be that no adjustments to it need to be made (e.g., when it is already a neutral TM with a reasonably high total traffic load). If the cost the sender is willing to incur can depend on the actual traffic, then the sender can set a goal of some minimum threshold of uncertainty on the part of the adversary as measured by the entropy of the observed traffic matrix, then try to achieve that entropy with minimum cost. If the sender has to live within a budget, then some average cost per period may be set as a goal, and the sender can try to maximize entropy within this average cost constraint. Here, there may be two variants:

- **Offline:** the sender knows what the traffic is going to be for many periods ahead of time, and can pick a cost for each period that balances the entropy that can be achieved for each period within its cost;
- **Online:** the sender only knows the amortized cost goal and the history of traffic and costs up until the current time.

In the offline case, the sender can achieve greater entropy if most of the actual TMs in the sequence have high entropy to begin with, or avoid having some observed TMs at the end of the sequence with low entropy because the budget was exhausted too early in the sequence.

Online computation will suffer from these possibilities, but the goals can be changed dynamically given the history and remaining budget, if there is any reason to believe that the future actual TMs can be predicted from the recent past TMs.

### 5.3 Sender and Adversary Knowledge

In the strongest case, the sender may know the sequence of $T_{act}(i)$'s, or at least the set (but not the order) ahead of time and be able to plan how to disguise that particular set of actual TMs. A weaker assumption is that the sender knows the probability distribution for the actual TMs (or for properties they possess) ahead of time, and the actual sequence is close to this (defined by some error metric).

What the adversary sees, and what the adversary knows, *a priori*, determine what the adversary learns from a sequence of observations. For example, if the sender always sends neutral TMs of the same magnitude the adversary learns

very little (only a bound on the total load), but the sender must accept whatever cost is needed to arrive at the neutral TM that is always sent.

On the other hand, if the sender sends different TMs each period, then what the adversary learns can depend on what the sender had to disguise and the adversary's knowledge of that.

For example, if the sender always has the same actual TM, but disguises it differently each time, and the adversary knows this, then that adversary can take the intersection of all of the sets of TMs consistent with the observed TMs over time to reduce uncertainty over what was actually sent:

$$T_{act} \in \cap_{i=1}^{k} \mathbb{T}_{T_{obs}}(i),$$

where $T_{obs}(i)$ is the $i^{th}$ observed TM. The entropy (if all TMs are equally probable) is then

$$S = lg(|\cap_{i=1}^{k} \mathbb{T}_{T_{obs}}(i)|),$$

where $lg$ is shorthand for $log_2$. Other adversary information (on sender cost budgets or expected traffic pattern properties) may further limit the entropy.

If the sender always uses the same cost $c$ for each period, and the adversary knows this cost, then as stated in section 5.2, the adversary knows that

$$T_{act} \in \{T \in \mathbb{T}_{T_{obs}} | d(T, T_{obs}) = c\}.$$

The entropy is then

$$S = lg(|\{T \in \mathbb{T}_{T_{obs}} | d(T, T_{obs}) = c\}|).$$

If the sender has different actual TMs each period, and has a cost distribution that is randomly applied (and the adversary knows what it is), then the adversary can determine the probability for each $T \in \mathbb{T}_{T_{obs}}$ according to $d(T, T_{obs})$.

Let

$$\mathbb{S}_c(T_{obs}) = \{T \in \mathbb{T}_{<G,w>} | d(T, T_{obs}) = c\}$$

be the hypersphere at distance $c$ from $T_{obs}$ of feasible traffic matrices for a graph $G$. Let

$$\mathbb{P}_c(T_{obs}) = \{T \in \mathbb{T}_{T_{obs}} | d(T, T_{obs}) = c\} = \mathbb{T}_{T_{obs}} \cap \mathbb{S}_c(T_{obs})$$

be the intersection of the hypersphere at distance $c$ from $T_{obs}$ and the TMs from which $T_{obs}$ can be $R, P$-derived, $\mathbb{T}_{T_{obs}}$. Let

$$U = \{(c, p_c)\}$$

be the sender's probability distribution for costs (i.e., cost $c$ is incurred with probability $p_c$). Of course this distribution is dependent on how we do our TAP, and should be considered as a dynamic distribution. So

$$\sum_{c=0}^{\infty} p_c = 1.$$

Then the attacker can infer that

$$\sum_{T \in \mathbb{P}_c(T_{obs})} prob(T|T_{obs}, U) = p_c, \qquad \text{so}$$

$$prob(T|T_{obs}, U) = \frac{p_c}{|\mathbb{P}_c(T_{obs})|} \qquad \text{for } T \in \mathbb{P}_c(T_{obs})^{10}.$$

If the sender adapts the cost to the actual traffic matrix, but still has an amortized cost per period goal that the adversary knows, then it may still be possible for the adversary to assign probabilities to the TMs in $\mathbb{T}_{T_{obs}}$ based on assumptions (or knowledge) of the nature of the distribution of the actual TMs.

## 6 Transforms

This section formally describes the two types of TAP method considered in this paper, padding and rerouting.

### 6.1 Padding

If we limit the TAP method to be padding only, then every element of $T_{act}$ is pointwise bounded by the corresponding element of $T_{obs}$:

$$T_{act}[i, j] \ \leq \ T_{obs}[i, j].$$

In fact,

$$T_{obs} \ = \ T_{act} + P,$$

where $P$ is a traffic matrix (i.e., it is non-negative) representing the pad traffic added to the true traffic in $T_{act}$.

### 6.2 Rerouting

If the TAP method is limited to rerouting alone, then the true traffic matrix must be a preimage of the apparent traffic matrix under transformation by some rerouting quantities. Rerouting effects will be represented by a rerouting difference matrix, $D_r$, that describes the change in traffic due to rerouting, so that

$$T_{obs} \ = \ T_{act} + D_r.$$

Note that $D_r$ may have negative elements.

For distinct nodes $A, B, C \in [1..N]$ we define the unit reroute matrix as follows. The unit reroute matrix $U_{A,B,C}$ for rerouting one unit of traffic from $A$

[10] There is a little hair here. The probability distribution may have a long tail (i.e., large $c$'s have nonzero $p_c$'s), but for a particular $T_{obs}$, there is a maximum possible distance for TMs in $\mathbb{P}_c(T_{obs})$. The adversary must normalize the distribution over the set of possible costs to account for this.

to $C$ via $B$ is the $N \times N$ matrix consisting of all zeros except that $U_{A,B,C}[A,C] = -1$, representing a unit decrease in the traffic from $A$ to $C$ due to rerouting, and $U_{A,B,C}[A,B] = U_{A,B,C}[B,C] = 1$, representing a unit increase in the traffic from $A$ to $B$ and from $B$ to $C$ due to rerouting.

$$U_{A,B,C}[i,j] = \begin{cases} 1 \text{ iff } (i = A \wedge j = B) \vee (i = B \wedge j = C) \\ \text{-1 iff } i = A \wedge j = C \\ 0 \text{ otherwise} \end{cases}$$

The unit reroute matrix $U_{A,B,C}$ has row and column sums equal to zero for all rows and columns except for the intermediate node's:

$$\sum_{i=1}^{N} U_{A,B,C}[i,j] = 0 \quad \forall j \in [1..N],\ j \neq B,$$

$$\sum_{j=1}^{N} U_{A,B,C}[i,j] = 0 \quad \forall i \in [1..N],\ i \neq B.$$

For the intermediate node, $B$, the row and column sum are each equal to one:

$$\sum_{i=1}^{N} U_{A,B,C}[i,B] = 1,$$

$$\sum_{j=1}^{N} U_{A,B,C}[B,j] = 0.$$

The total change in the traffic load due to a unit reroute is thus one.

Reroute quantities may be represented by a 3-dimensional array, $r[A,B,C]$, indicating the number of packets rerouted from source $A$ via intermediate node $B$ to destination $C$. Note that the reroute quantities $r[A,A,A]$, $r[A,A,B]$, and $r[A,B,B]$ are all zero, as they represent either self-communication or rerouting via either the source or destination node itself.

From the reroute quantities and the unit reroute matrices, we may compute the rerouting difference matrix, $D_r$, which represents the net rerouting effects for all rerouting specified by $r$ simultaneously. If $k$ units of traffic are rerouted from $A$ to $C$ via $B$, then a contribution of $k\,U_{A,B,C}$ is made by these rerouted packets to $D_r$. Then the matrix representing the net difference due to rerouting is just the elementwise matrix sum of the weighted unit reroute matrices,

$$D_r \ = \sum_{A,B,C \in [1..N]} r[A,B,C]\, U_{A,B,C}$$

Any rerouting difference matrix $D_r$ of a non-negative $r$ must have a non-negative sum over all its elements (or aggregate traffic load), in fact,

$$\sum_{i=1}^{N}\sum_{j=1}^{N} D_r[i,j] \ = \ \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N} r[i,j,k].$$

Since each unit reroute matrix represents a unit increase in the total traffic load, it is obvious that the total increase in the aggregate traffic load is equal to the total amount of rerouting performed.

### 6.3 Discussion

Both padding and rerouting cause a net increase in the resultant TM. Thus, for a TM $T$ to be a preimage of an observed TM, $T_{obs}$, its total load is bounded above by the total load of the observed TM,

$$L(T) \leq L(T_{obs}) \ .$$

Furthermore, it may be noted that for both transforms, the row and column totals either remain the same or increase. Therefore,

$$\sum_{i=1}^{N} T[i,j] \ \leq \ \sum_{i=1}^{N} T_{obs}[i,j] \ \forall \ j \in [1..N], \qquad \text{and}$$

$$\sum_{j=1}^{N} T[i,j] \ \leq \ \sum_{j=1}^{N} T_{obs}[i,j] \ \forall \ i \in [1..N], \qquad \text{for any } T \in \mathbb{T}_{T_{obs}}.$$

An arbitrary $N \times N$ matrix whose sum of elements is non-negative may not be realizable as a rerouting difference matrix. There may be negative elements in the rerouting difference matrix, so the true traffic matrix $T_{act}$ is not constrained to be pointwise bounded by $T_{obs}$, as is the case when only padding was used. However, the row and column traffic bounds and the constraints on the rerouting difference matrices do limit the set of traffic matrices that could give rise to an observed TM. This in turn means that for some TM's, the conditional probability will be zero for a given $T_{obs}$ even if the aggregate traffic bound, or even the row and column traffic constraints are satisfied.

Now the issue is the degree to which the uncertainty that can be created by rerouting and padding is adequate to mask the true TM. This is in effect represented by the entropy.

## 7 Examples

Consider a simple example – the attacker observes 3 nodes sending 1 message to each other, but, of course, not to themselves. She knows nothing about the padding or rerouting policies of these nodes. Let us see what level of anonymity this gives us. The observed matrix is:

$$T_{obs} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}.$$

The rows (columns) represent a message leaving (going to) nodes $A$, $B$, or $C$ respectively. We now try to calculate the set of $T_{obs}$ which could have resulted in the above $T_{act}$ after having been subjected to padding or rerouting.

We start by considering rerouting. There are six possible traffic matrices that can be rerouted into $T_{obs}$. Consider $T_1 = \begin{pmatrix} 0 & 2 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$. If we take one message that was sent from $A$ to $B$, and redirect that message via the intermediary node $C$, our new traffic matrix is just $T_{obs}$. Thus, we see that rerouting can hide the true traffic pattern, which is $T_1$, by making the traffic pattern look like $T_{obs}$. In fact there are five more traffic matrices which can be disguised to look like $T_{obs}$ by using one rerouting of a message. Those traffic matrices are $T_2, \ldots, T_6$ $= \begin{pmatrix} 0 & 0 & 2 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 1 \\ 2 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 2 \\ 1 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 2 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 2 & 0 \end{pmatrix}$.

Now consider rerouting two messages. Observe the matrix $T_{-,1} = \begin{pmatrix} 0 & 2 & 0 \\ 2 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$.

If that is the true traffic matrix, then we can disguise this traffic pattern by taking one of the messages from B to A, and redirect it through C, this results in the above traffic matrix $T_1$, and as we noted another rerouting at this level will result in $T_{obs}$. But notice that $T_{-,1}$ will also result in $T_3$ after rerouting on one of the A to B messages through C. Therefore, we see that this second level inverse rerouting result in three unique traffic matrices. At this point we see there are $6 + 3 = 9$ possible traffic matrices that are hidden by $T_{obs}$.

We have been concentrating on rerouting. Let us now turn our attention to padding. The traffic after the padding has been applied must equal $T_{obs}$, so each link can be padded by at most 1 message. This gives us six entries in the matrix with the freedom of one bit for each entry. This results in $2^6$ possible traffic matrices. Since we count $T_{obs}$ itself as a possible traffic matrix this gives us $2^6 - 1$ additional traffic matrices.

So far, we have 1 traffic matrix if we count $T_{obs}$, another $2^6 - 1$ by counting possible traffic matrices by padding, 6 by counting rerouting of 1 message, and another 3, by counting a prior rerouting. We are not done yet. Consider the six traffic matrices $T_1, \ldots, T_6$ that results from rerouting of 1 message. Each one of these may be the result of padding from a sparser traffic matrix. For example consider $T_2$ and the lower triangular entries that are ones. If the original traffic matrix was $\begin{pmatrix} 0 & 0 & 2 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ we can obtain $T_2$ by two 1-pads. In fact we see that the entries that "are one" in $T_2$ give us three degrees of freedom, with one bit for each degree of freedom. This results in $2^3$ possible traffic matrices that result into $T_2$ after the 1-pads. So as not to count $T_2$ twice this gives us $2^3 - 1$ unique traffic matrices. This follows for all six of the one-level rerouting traffic matrices. Therefore, we have an additional $6(2^3 - 1)$ possible traffic matrices to consider.

So we see that $|\mathbb{T}_{T_{obs}}| = 1+(2^6-1)+6(2^3-1)+6+3 = 2^6+3(2^4+1) = 115$. This hides the actual traffic matrix behind a probabilistic value of 1/115. If $T_{obs}$ was a little more exciting, say it was $\begin{pmatrix} 0 & 5 & 5 \\ 5 & 0 & 5 \\ 5 & 5 & 0 \end{pmatrix}$, the probability of the actual traffic matrix would be much smaller, but this lower probability comes at the cost of excessive reroutes and padding. Therefore, pragmatic choices must be made, as is usually the case, when one wishes to obfuscate their true business on a network.

## 8 Conclusions

This paper represents a step in the direction of precisely defining the amount of success a TAP system has in hiding the nature of the actual traffic matrix from a global, passive adversary. Padding and rerouting are considered, with observations on the effects each has on the difference between the actual and the observed TM. The paper introduces an entropy-based approach to the amount of uncertainty the adversary has in determining the actual TM, or alternatively, the probability that the actual TM has a property of interest.

If the sender has no cost constraints, then it may adopt a strategy of transmitting neutral TMs, providing the adversary with minimal information. If the sender does have cost constraints, then it may not be able always to send neutral TMs, so it must use other approaches. The goal may be to maintain a certain cost distribution and to maximize the adversary's uncertainty within that budget, or it may be to achieve a minimum degree of uncertainty in the adversary while minimizing the cost of doing so.

## Acknowledgements

We thank the anonymous reviewers for helpful comments and suggestions. Andrei Serjantov acknowledges the support of EPSRC research grant GRN24872 Wide Area programming and EC FET-GC IST-2001-33234 PEPITO project. Ira Moskowitz, Richard Newman, and Paul Syverson were supported by ONR.

## References

1. Adam Back, Ulf Möller, and Anton Stiglic. Traffic analysis attacks and trade-offs in anonymity providing systems. In Ira S. Moskowitz, editor, *Information Hiding, 4th International Workshop (IH 2001)*, pages 245–257. Springer-Verlag, LNCS 2137, 2001.
2. O. Berthold and H. Langos. Dummy traffic against long term intersection attacks. In Paul Syverson and Roger Dingledine, editors, *Privacy Enhancing Technologies (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.
3. Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In Paul Syverson and Roger Dingledine, editors, *Privacy Enhancing Technologies (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.

4. Leonidas Georgiadis, Roch Guérin, Vinod Peris, and Kumar N. Sivarajan. Efficient network QoS provisioning based on per node traffic shaping. *IEEE/ACM Transactions on Networking*, 4(4):482–501, 1996.
5. D. Goldschlag, M. Reed, and P. Syverson. Hiding routing information. In Ross Anderson, editor, *Information Hiding, First International Workshop*, pages 137–150. Springer-Verlag, LNCS 1174, May 1996.
6. F. Halsall. *Data Communications, Computer Networks, and Open Systems*. Addison-Wesley, 1992.
7. Myong H. Kang, Ira S. Moskowitz, and Daniel C. Lee. A network Pump. *IEEE Transactions on Software Engineering*, 22(5):329–328, 1998.
8. R. E. Newman-Wolfe and B. R. Venkatraman. High level prevention of traffic analysis. In *Proc. IEEE/ACM Seventh Annual Computer Security Applications Conference*, pages 102–109, San Antonio, TX, Dec 2-6 1991. IEEE CS Press.
9. R. E. Newman-Wolfe and B. R. Venkatraman. Performance analysis of a method for high level prevention of traffic analysis. In *Proc. IEEE/ACM Eighth Annual Computer Security Applications Conference*, pages 123–130, San Antonio, TX, Nov 30-Dec 4 1992. IEEE CS Press.
10. Onion routing home page. `http://www.onion-router.net`.
11. Andreas Pfitzmann and Marit Köhntopp. Anonymity, unobservability and pseudonymity — a proposal for terminology. In Hannes Federrath, editor, *Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Observability*, pages 1–9. Springer-Verlag, LNCS 2009, July 2000.
12. Charles Rackoff and Daniel R. Simon. Cryptographic defense against traffic analysis. In *ACM Symposium on Theory of Computing*, pages 672–681, 1993.
13. J. Raymond. Traffic analysis: Protocols, attacks, design issues, and open problems. In Hannes Federrath, editor, *Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Observability*, pages 10–29. Springer-Verlag, LNCS 2009, July 2000.
14. Michael K. Reiter and Aviel D. Rubin. Crowds: anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.
15. Andrei Serjantov and George Danezis. Towards an information theoretic metric for anonymity. In Paul Syverson and Roger Dingledine, editors, *Privacy Enhacing Technologies (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.
16. C.E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28:656–715, 1949.
17. W. Stallings. *Data and Computer Communications (6th Ed.)*. Prentice-Hall, 2000.
18. Paul Syverson and Stuart Stubblebine. Group principals and the formalization of anonymity. In J.M. Wing, J. Woodcock, and J. Davies, editors, *FM'99 – Formal Methods, Vol. I*, pages 814–833. Springer-Verlag, LNCS 1708, March 1999.
19. Paul F. Syverson, Gene Tsudik, Michael G. Reed, and Carl E. Landwehr. Towards an analysis of onion routing security. In Hannes Federrath, editor, *Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Observability*, pages 96–114. Springer-Verlag, LNCS 2009, July 2000.
20. B. R. Venkatraman and R. E. Newman-Wolfe. Performance analysis of a method for high level prevention of traffic analysis using measurements from a campus network. In *Proc. IEEE/ACM Tenth Annual Computer Security Applications Conference*, pages 288–297, Orlando, FL, December 5-9 1994. IEEE CS Press.
21. B. R. Venkatraman and R. E. Newman-Wolfe. Capacity estimation and auditability of network covert channels. In *Proc. IEEE Symposium on Security and Privacy*, pages 186–198, Oakland, CA, May 8-10 1995. IEEE CS Press.

# Breaking and Mending Resilient Mix-Nets

Lan Nguyen and Rei Safavi-Naini

School of Information Technology and Computer Science
University of Wollongong
Wollongong 2522, Australia
{ldn01,rei}@uow.edu.au

**Abstract.** In this paper we show two attacks against universally resilient mix-nets. The first attack can be used against a number of mix-nets, including Furukawa-Sako01 [6], Millimix [11], Abe98 [1], MiP-1, MiP-2 [2,3] and Neff01 [19]. We give the details of the attack in the case of Furukawa-Sako01 mix-net. The second attack breaks the correctness of Millimix [11]. We show how to counter these attacks, and give efficiency and security analysis for the proposed countermeasures.

## 1 Introduction

Mix-net [4] is a cryptographic technique that is used to hide the origin of messages in network communication. It can be used in a wide range of applications, including anonymous email [4], Web browsing [7], electronic voting [22,14], anonymous payment systems [9], secure multiparty computation [12] and privacy in advertisements [15]. A mix-net consists of a set of mix servers, each receiving as input a list of ciphertexts and outputting either a permuted list of the re-encrypted ciphertexts, or a permuted list of the corresponding plaintexts. By keeping the permutation secret, the mix-net can hide the correspondence between input items and output items hence providing *privacy* for the originators of messages. Other important properties of mix-nets are *robustness* and *verifiability*. Robustness means that the mix-net is able to operate correctly regardless of component failure. Verifiability means that the correctness of mix-net operation can be verified by any system participant. A mix-net that provides privacy, robustness and verifiability is called *resilient* [5].

A common way of proving correctness of the results is that each mix-server after producing the output, performs a *verification protocol*. If the verification results in *accept*, the mix-server is assumed *honest* and its output is taken to the next mix-server. If the verification outputs *reject*, the mix-server is found *dishonest* and is expelled from the mix-net and its input is passed to the next mix-server. Many mix-nets, including those given in [11,2,3,6,19] use zero-knowledge proofs based on the difficulty of *discrete logarithm problem* to perform their verification protocol.

In this paper we show that two mix-nets, proposed by Furukawa et al. [6] and Jakobsson et al. [11], are not resilient. We show that in both cases, despite provable security, a mix-server can produce incorrect output without being detected.

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 66–80, 2003.

One of the attacks can also be used against other mix-nets, including mixing phase in MiP-1, MiP-2 [2,3] and Neff01 [19] and decryption phase in Abe98 [1] and MiP-2 [2,3].

The organization of the paper is as follows. In section 2, we recall cryptographic tools and systems that will be used in the rest of the paper. Section 3 gives brief descriptions of Furukawa-Sako01 and Millimix mix-nets. The next two sections show attacks, countermeasures and analysis for Furukawa-Sako01 and Millimix mix-nets. Section 6 concludes the paper.

## 2 Background

### 2.1 Model

A mix-net consists of the following participants that are all assumed polynomially bounded. *Users* send messages to mix-net. *Mix servers* perform mixing of then input messages and produce an output, which could be used as input to other mix-servers. Verification can be *external* where a trusted *verifier* verifies operation of the mix-net, or *internal* where each mix server is verified by other mix servers in the same mix-net. We assume there is a *bulletin board* which is a shared memory where all participants have read access to and can append messages after being authenticated. A bulletin board simulates an authenticated broadcast channel.

An *adversary* is a party whose objective is to compromise resiliency of the mix-net. An adversary that can corrupt $t_u$ users and $t_s$ mix servers is called a $(t_u, t_s)$ adversary. Corruption is before the system starts operation (*static adversary*).

### 2.2 Requirements

To define resiliency we follow the definitions in [5].

- *privacy:* it is infeasible for the adversary to output a pair of input and the corresponding output of an honest user with probability significantly greater than random guess.
- *verifiability:* if a set of participating mix servers produce an output different from the one prescribed by the protocol, then the verification will be able to establish this fact and reveal the identities of the cheating servers. If verification only uses publicly available information of the mix-net, the mix-net is called *universally verifiable*.
- *robustness:* ensures that the probability of producing incorrect output is negligibly less than 1.
- *efficiency* is measured in terms of the computation and communication costs of participants.

A mix-net is *resilient* if it satisfies *privacy*, *robustness* and *verifiability*. A resilient mix-net is *universally resilient* if it is universally verifiable. It is *optimally resilient* if $t_u$ and $t_s$ have their maximum possible values, that is $t_u$ is equal $n-2$ where $n$ is he number of users, and $t_s$ is equal to $\lfloor (s-1)/2 \rfloor$ in case of *internal verification* and is equal to $s-1$ in the case of *external verification*.

### 2.3 Cryptographic Tools

**El Gamal-Schnorr Non-malleable Encryption.** Inputs to a mix-net must be encrypted by a *non-malleable* encryption scheme. In a non-malleable encryption scheme, given a ciphertext it is computationally infeasible to generate a different ciphertext such that the corresponding plaintexts are related in a known manner. If the encryption scheme is malleable, an adversary can trace an input ciphertext $ci$, by creating a ciphertext $ci'$ whose plaintext is related to the plaintext of $ci$ in a known manner and in the output check for the plaintexts that satisfy the relationship. An example of this attack is shown in [23] against mix-net in [22]. Most of mix-net schemes use a combination of El Gamal encryption and Schnorr signature to efficiently achieve non-malleability. El Gamal-Schnorr non-malleable encryption scheme [2] can be described as follows. Let $p$ and $q$ be two large primes such that $p = 2kq + 1$, where k is a positive integer and $g$ is a generator of a subgroup $G_q$ of order q in $Z_p^*$. Hereafter, unless stated otherwise we assume all computations are in modulo $p$. The private key is $x \in Z_q$ and the public key is $(y, g)$ where $y = g^x$. A ciphertext of message $m \in G_q$ is $(\alpha, \beta, c, z)$ where $\alpha = my^s$, $\beta = g^s$, $c = H(\alpha, \beta, g^w)$, $z = w - cs \bmod q$, $H$ is a hash function $H : \{0,1\}^* \rightarrow 2^{|q|}$ and $s, w \in_R Z_q$ (i.e. chosen randomly and with uniform distribution from $Z_q$). Validity of a ciphertext can be verified by checking whether $c \stackrel{?}{=} H(\alpha, \beta, g^z\beta^c)$ and $\alpha, \beta \in G_q$. Intuitively, Schnorr signature is used to show that the ciphertext must have been encrypted by someone with the knowledge of $s$. The plaintext is computed as $m := \alpha/\beta^x$.

An El Gamal-only ciphertext $(\alpha, \beta)$ can be *re-encrypted* as another ciphertext $(\alpha \times y^r, \beta \times g^r)$ of the same plaintext $m$, where *re-encryption exponent* $r \in_R Z_q$.

**Schnorr Identification.** Let $p$,$q$,$x$ and $(y, g)$ be defined as above. A prover $\mathcal{P}$ can show his knowledge of the private key $x$ to a verifier $\mathcal{V}$ using Schnorr identification protocol as follows.

1. $\mathcal{P} \longrightarrow \mathcal{V}$: a commitment $w = g^e$, where $e \in_R Z_q$
2. $\mathcal{P} \longleftarrow \mathcal{V}$: a challenge $c \in_R Z_q$
3. $\mathcal{P} \longrightarrow \mathcal{V}$: a response $s = e + cx \bmod q$

$\mathcal{V}$ then verifies that $g^s = wy^c$. Schnorr identification protocol can be converted into a Schnorr signature scheme by generating $c = H(w, m)$ for a message $m$ that is to be signed using a hash function $H : \{0,1\}^* \rightarrow 2^{|q|}$. Schnorr signature is used in the encryption scheme above.

**Disjunctive Schnorr Identification.** Let $p$ and $q$ be defined as above. Suppose $(x_1, (y_1, g_1))$ and $(x_2, (y_2, g_2))$ are two instantiations of $(x, (y, g))$ above. A prover $\mathcal{P}$ shows he has one of the private keys $x_1$ or $x_2$ to a verifier $\mathcal{V}$ by using the *Disjunctive Schnorr identification protocol* as follows. Assume that $\mathcal{P}$ possesses $x_1$.

1. $\mathcal{P} \longrightarrow \mathcal{V}$: two commitments $w_1 = g_1^{e_1}$, $w_2 = g_2^{s_2} y_2^{-c_2}$, where $e_1, e_2, c_2, s_2 \in_R Z_q$
2. $\mathcal{P} \longleftarrow \mathcal{V}$: a challenge $c \in_R Z_q$
3. $\mathcal{P} \longrightarrow \mathcal{V}$: responses $s_1 = e_1 + c_1 x_1 \bmod q$, $s_2$, $c_1 = c \oplus c_2$, $c_2$

$\mathcal{V}$ then checks if $g_i^{s_i} = w_i y_i^{c_i}$ for $i \in \{1, 2\}$. Similar to Schnorr identification protocol, Disjunctive Schnorr identification protocol can be converted into a non-interactive form.

**Permutation Matrices.** A matrix $(A_{ij})_{n \times n}$ is a permutation matrix if there exists a permutation $\phi$ so that $\forall i, j \in \{1, ..., n\}$

$$A_{ij} = \begin{cases} 1 \bmod q \text{ if } \phi(i) = j \\ 0 \bmod q \text{ otherwise} \end{cases}$$

It is proved in [6] that an integer valued matrix $(A_{ij})_{n \times n}$ is a permutation matrix if and only if $\forall i, j, k \in \{1, ..., n\}$

$$\sum_{h=1}^{n} A_{hi} A_{hj} = \begin{cases} 1 \bmod q \text{ if } i = j \\ 0 \bmod q \text{ otherwise} \end{cases} \quad (1)$$

$$\sum_{h=1}^{n} A_{hi} A_{hj} A_{hk} = \begin{cases} 1 \bmod q \text{ if } i = j = k \\ 0 \bmod q \text{ otherwise} \end{cases} \quad (2)$$

**Pairwise Permutation Network.** Abe [2,3] used permutations that were constructed from switching gates. A *permutation network* is a circuit, which, on input $(1, ..., n)$ and an arbitrary permutation $\Pi : \{1, ..., n\} \rightarrow \{1, ..., n\}$, outputs $(\Pi(1), ..., \Pi(n))$. A *switching gate* is a permutation network for two input items. A *pairwise permutation network* [25] is a permutation network that is constructed from switching gates and requires $n \log_2 n - n + 1$ switching gates.

## 3 Mix-Nets

In this section we recall two mix-nets that are subjected to the attacks proposed in this paper.

### 3.1 Furukawa-Sako01 Mix-Net

**Overview.** This is one of the two most efficient mix-nets with optimal universal resiliency. Let $p$, $q$, private key $x$ and public key $(y, g)$ be set as above, with $p = 2kq + 1$, where k is a positive integer. Input to a mix-server is a set of El Gamal ciphertexts $\{(g_i, m_i) | i = 1, ..., n\}$ encrypted by the public key $(y, g)$, and so $g_i, m_i \in G_q$, $i = 1, ..., n$. A mix-server uses a permutation $\phi$ and re-encryption exponents $\{r_i | i = 1, ..., n\}$ to compute its output $\{(g'_i, m'_i) | i = 1, ..., n\}$ as follows:

$$g'_i = g^{r_i} g_{\phi^{-1}(i)}$$
$$m'_i = y^{r_i} m_{\phi^{-1}(i)}$$

To prove the correctness of its operation, the mix-server needs to show the existence of a permutation matrix $(A_{ij})_{n\times n}$ and $\{r_i|i=1,...,n\}$ so that:

$$g'_i = g^{r_i}\prod_{j=1}^{n} g_j^{A_{ji}} \tag{3}$$

$$m'_i = y^{r_i}\prod_{j=1}^{n} m_j^{A_{ji}} \tag{4}$$

This can be done by a verification protocol below that proves the following statements:

- Given $\{g_i\}$ and $\{g'_i\}$, $\{g'_i\}$ can be expressed as equation (3) using $\{r_i\}$ and a matrix that satisfies equation (1).
- Given $\{g_i\}$ and $\{g'_i\}$, $\{g'_i\}$ can be expressed as equation (3) using $\{r_i\}$ and a matrix that satisfies equation (2).
- The matrix and $\{r_i\}$ in the above two statements are the same.
- For each pair $(g'_i, m'_i)$, the same $r_i$ and $\{A_{ij}\}$ has been used.

**Verification.** The input is $p, q, g, y, \tilde{g}, \{\tilde{g_i}\}, \{(g_i, m_i)\}, \{(g'_i, m'_i)\}$, $i = 1, ..., n$, where $\{\tilde{g}, \tilde{g_1}, ..., \tilde{g_n}\}$ is a basis generated randomly and independently from the input ciphertexts, so that, under discrete logarithm assumption, it is computationally infeasible to obtain $\{a_i\}$ and $a$ satisfying $\tilde{g}^a \prod_{i=1}^{n} \tilde{g_i}^{a_i} = 1$. The prover is $\mathcal{P}$ and the verifier is $\mathcal{V}$.

1. $\mathcal{P}$ generates: $\delta, \rho, \tau, \alpha, \alpha_i, \lambda, \lambda_i \in_R Z_q, i = 1, ..., n$
2. $\mathcal{P}$ computes:

$$t = g^{\tau}, v = g^{\rho}, w = g^{\delta}, u = g^{\lambda}, u_i = g^{\lambda_i}, i = 1, ..., n$$

$$\tilde{g_i}' = \tilde{g}^{r_i}\prod_{j=1}^{n} \tilde{g_j}^{A_{ji}}, i = 1, ..., n \tag{5}$$

$$\tilde{g}' = \tilde{g}^{\alpha}\prod_{j=1}^{n} \tilde{g_j}^{\alpha_j} \tag{6}$$

$$g' = g^{\alpha}\prod_{j=1}^{n} g_j^{\alpha_j}$$

$$m' = y^{\alpha}\prod_{j=1}^{n} m_j^{\alpha_j}$$

$$\dot{t_i} = g^{\sum_{j=1}^{n} 3\alpha_j A_{ji} + \tau\lambda_i}, i = 1, ..., n$$

$$\dot{v_i} = g^{\sum_{j=1}^{n} 3\alpha_j^2 A_{ji} + \rho r_i}, i = 1, ..., n$$

$$\dot{v} = g^{\sum_{j=1}^{n} \alpha_j^3 + \tau\lambda + \rho\alpha}$$

$$\dot{w_i} = g^{\sum_{j=1}^{n} 2\alpha_j A_{ji} + \delta r_i}, i = 1, ..., n$$

$$\dot{w} = g^{\sum_{j=1}^{n} \alpha_j^2 + \delta\alpha}$$

3. $\mathcal{P} \longrightarrow \mathcal{V}$: $t, v, w, u, \{u_i\}, \{\tilde{g_i}'\}, \tilde{g}', g', m', \{\dot{t_i}\}, \{\dot{v_i}\}, \dot{v}, \{\dot{w_i}\}, \dot{w}, i = 1, ..., n$
4. $\mathcal{P} \longleftarrow \mathcal{V}$: challenges $\{c_i | i = 1, ..., n\}$, $c_i \in_U Z_q$
5. $\mathcal{P} \longrightarrow \mathcal{V}$:

$$s = \sum_{j=1}^{n} r_j c_j + \alpha$$

$$s_i = \sum_{j=1}^{n} A_{ij} c_j + \alpha_i \bmod q, i = 1, ..., n$$

$$\lambda' = \sum_{j=1}^{n} \lambda_j c_j^2 + \delta \bmod q$$

6. $\mathcal{V}$ verifies:

$$\tilde{g}^s \prod_{j=1}^{n} \tilde{g_j}^{s_j} = \tilde{g}' \prod_{j=1}^{n} \tilde{g_j}'^{c_j} \tag{7}$$

$$g^s \prod_{j=1}^{n} g_j^{s_j} = g' \prod_{j=1}^{n} g_j'^{c_j} \tag{8}$$

$$y^s \prod_{j=1}^{n} m_j^{s_j} = m' \prod_{j=1}^{n} m_j'^{c_j} \tag{9}$$

$$g^{\lambda'} = u \prod_{j=1}^{n} u_j^{c_j^2}$$

$$t^{\lambda'} v^s g^{\sum_{j=1}^{n} (s_j^3 - c_j^3)} = \dot{v} \prod_{j=1}^{n} \dot{v_j}^{c_j} \dot{t_j}^{c_j^2}$$

$$w^s g^{\sum_{j=1}^{n} (s_j^2 - c_j^2)} = \dot{w} \prod_{j=1}^{n} \dot{w_j}^{c_j}$$

### 3.2 Millimix

**Overview.** Millimix is a mix-net for small input batches that provides optimal universal resiliency with internal verification. Millimix uses El Gamal scheme for encryption. The two primes $p$ and $q$, private key $x$ and public key $(y, g)$ are set up as described above, and $p = 2q + 1$. To satisfy non-malleability, El Gamal-Schnorr non-malleable encryption scheme can be used. Input to the mix-net is a set of ciphertexts encrypted by the public key $(y, g)$. If El Gamal-Schnorr non-malleable encryption scheme is used, and an input ciphertext $(\alpha, \beta, c, z)$ needs to pass non-malleability test before $(\alpha, \beta)$ is taken to the first mix-server. Each mix-server, except the first one, takes output of the previous mix-server as its input, and the output of the mix-net is the output of the last mix server.

Each mix server simulates a pairwise permutation network consisting of a number of switching gates. Each switching gate re-encrypts and permutes the two

input ciphertexts. The mix server outputs the result of permutation and proves the correctness of each of its switching gate's operations using a verification protocol described in the following section. Once a corrupt mix server is found, it is expelled and its input is passed to the next mix server. If the corrupt mix-server is the last mix-server, its input is posted to the bulletin board as the output of the mix-net.

The system is efficient because for an input batch of $n$ items, each mix server needs $O(nlogn)$ modular exponentiations with low constant to perform the re-encryption and internal verification.

Millimix uses threshold decryption to decrypt the input list of ciphertexts. We omit the details of this as it is not relevant to the attack described below, which breaks the correctness of the system.

**Verification.** The verification is by proving correctness of the output of each switching gate. The input to a switching gate is a pair of ciphertexts $(\alpha_1, \beta_1)$, $(\alpha_2, \beta_2)$ of the two plaintexts $m_1$, $m_2$ respectively, and the output is a pair of ciphertexts $(\alpha'_1, \beta'_1)$, $(\alpha'_2, \beta'_2)$ of the two plaintexts $m'_1$, $m'_2$ respectively. The server shows its correctness of the switching gate by proving the following two statements:

- Statement 1: $m_1 m_2 = m'_1 m'_2$ using Plaintext Equivalent Proof ($PEP$) for ciphertexts $(\alpha_1\alpha_2, \beta_1\beta_2)$ and $(\alpha'_1\alpha'_2, \beta'_1\beta'_2)$.
- Statement 2: $m_1 = m'_1$ OR $m_1 = m'_2$ using DISjunctive Plaintext Equivalent Proof ($DISPEP$)

$PEP$ proves a ciphertext $(\alpha', \beta')$ is a valid re-encryption of a ciphertext $(\alpha, \beta)$ encrypted using El Gamal public key $(y, g)$. That is there exists $\gamma \in Z_q$ such that $\alpha = \alpha' y^\gamma$ and $\beta = \beta' g^\gamma$. Jakobsson et al [11] showed that this proof can be obtained using Schnorr identification protocol (or Schnorr signature for non-interactive case) as described below. Assume two ciphertexts $(\alpha, \beta)$ and $(\alpha', \beta')$ are given, compute $(y_s, g_s) = ((\alpha/\alpha')^z(\beta/\beta'), y^z g)$. Now if $(\alpha, \beta)$ and $(\alpha', \beta')$ are encryptions of the same message, then there exists $\gamma \in Z_q$ such that $(y_s, g_s) = ((y^z g)^\gamma, y^z g)$. The prover (mix-server) uses Schnorr identification protocol to show that it knows $\gamma$.

$DISPEP$ proves that a ciphertext $(\alpha_1, \beta_1)$ represents a re-encryption of one of the two ciphertexts $(\alpha'_1, \beta'_1)$ and $(\alpha'_2, \beta'_2)$. $DISPEP$ is implemented by having the prover to perform Disjunctive Schnorr identification protocol. Jakobsson et al. suggested to find $(y_{s1}, g_{s1}) = (\alpha_1/\alpha'_1, \beta_1/\beta'_1)$ and $(y_{s2}, g_{s2}) = (\alpha_1/\alpha'_2, \beta_1/\beta'_2)$ as two valid Schnorr public keys and use Disjunctive Schnorr identification protocol to show knowledge of one of the Schnorr private keys, which is also the El Gamal private key $x$ of the ciphertexts. Therefore, this requires the mix-server to know the El Gamal private key $x$ of the ciphertexts, which is not acceptable. In section 4.2, we show a revised version of this protocol which uses the approach in $PEP$ and removes this problem.

# 4 Attacks

In this section we propose two attacks that break the resiliency of a number of mix-nets. We describe the first attack on Furukawa-Sako01 scheme and comment on its application to other schemes.

## 4.1 Attacking Furukawa-Sako01 Scheme

**Description.** It is possible to break correctness of this mix-net with a success chance of at least 50%.

Let $a$ be a generator of $Z_p$. Then $a^{kq} \neq 1$ and $a^{2kq} = 1$. The mix server modifies one of the output ciphertexts as

$$g'_{i_0} = g^{r_{i_0}} g_{\phi^{-1}(i_0)}$$
$$m'_{i_0} = y^{r_{i_0}} m_{\phi^{-1}(i_0)} a^{kq}$$

where $i_0 \in \{1, ..., n\}$. $(g'_{i_0}, m'_{i_0})$ is not a valid re-encryption of $(g_{\phi^{-1}(i_0)}, m_{\phi^{-1}(i_0)})$. However, if the challenge $c_{i_0}$ is even, then the verification protocol still accepts the output as correct, as shown below.

The mix server only modifies $m'_{i_0}$ which only affects equation (9) in the verification protocol. If the verifier can successfully verify equation (9), the verification protocol produces incorrect results. Because $c_{i_0}$ is even, $a^{c_{i_0} kq} = 1$. So

$$m'^{c_{i_0}}_{i_0} = (y^{r_{i_0}} m_{\phi^{-1}(i_0)} a^{kq})^{c_{i_0}} = (y^{r_{i_0}} m_{\phi^{-1}(i_0)})^{c_{i_0}}$$

Therefore, equation (9) remains correct.

In another version of this attack, the mix server modifies $g'_{i_0}$ in a similar manner so that the incorrect ciphertext becomes

$$g'_{i_0} = g^{r_{i_0}} g_{\phi^{-1}(i_0)} a^{kq}$$
$$m'_{i_0} = y^{r_{i_0}} m_{\phi^{-1}(i_0)}$$

**Countermeasure.** Multiplying $y^{r_{i_0}} m_{\phi^{-1}(i_0)}$ by $a^{kq}$ generates a $m'_{i_0}$ that is not in $G_q$. The attack can be detected if the verifier checks to see if $g'_i, m'_i \in G_q$, $i = 1, ..., n$ . If $k = 1$, it is the same as checking the Legendre symbol of $g'_i, m'_i$, for which an algorithm can be found in [16] (p. 73). The algorithm requires one extra modular multiplication.

If $k \neq 1$, two extra modular exponentiations are required. So the verification cost at each mix server will increase by $2n$ modular exponentiations, where $n$ is the number of input items.

**Security.** Furukawa-Sako01 protocol has been proved to be complete, sound and zero-knowledge. In the following, we show the effect of the above attack on the proof of soundness and note that completeness and zero-knowledgeness proofs will not be affected by the proposed attack.

The proof of soundness is based on Lemma 1 restated from [6]. We show the short-coming of the original proof of the lemma and how the proposed fix completes the proof.

**Lemma 1.** *Assume $\mathcal{P}$ knows $\{A_{ij}\}, \{r_i\}, \{\alpha_i\}$ and $\alpha$ satisfying equations (5) and (6), and $\{s_i\}$ and $s$ satisfying equation (7). If equations (8) and (9) hold with non-negligible probability, then either the relationships*

$$\begin{cases} g' &= g^{\alpha} \prod_{j=1}^{n} g_j^{\alpha_j} \\ g'_i &= g^{r_i} \prod_{j=1}^{n} g_j^{A_{ji}}, i = 1, ..., n \\ m' &= y^{\alpha} \prod_{j=1}^{n} m_j^{\alpha_j} \\ m'_i &= y^{r_i} \prod_{j=1}^{n} m_j^{A_{ji}}, i = 1, ..., n \end{cases}$$

*hold or $\mathcal{P}$ can generate nontrivial integers $\{a_i\}$ and $a$ satisfying $\tilde{g}^a \prod_{i=1}^{n} \tilde{g_i}^{a_i} = 1$ with overwhelming probability.*

*Proof.* Replace $\tilde{g'}$ and $\{\tilde{g'_i}\}$ in equation (7) by those corresponding values in equation (5) and (6), we have the following:

$$\tilde{g}^{\sum_{j=1}^{n} r_j c_j + \alpha - s} \prod_{i=1}^{n} \tilde{g_i}^{\sum_{j=1}^{n} A_{ij} c_j + \alpha_i - s_i} = 1$$

Therefore, either the equations

$$\begin{cases} s = \sum_{j=1}^{n} r_j c_j + \alpha \\ s_i = \sum_{j=1}^{n} A_{ij} c_j + \alpha_i \end{cases}$$

hold or $\mathcal{P}$ can generate nontrivial integers $\{a_i\}$ and $a$ satisfying $\tilde{g}^a \prod_{i=1}^{n} \tilde{g_i}^{a_i} = 1$ with overwhelming probability.

If the equations hold, replace $s$ and $\{s_i\}$ in equation (8), we have the following equation holds with non-negligible probability

$$1 = b_0 \prod_{i=1}^{n} b_i^{c_i} \tag{10}$$

where

$$b_0 = \frac{g^{\alpha} \prod_{j=1}^{n} g_j^{\alpha_j}}{g'}$$

$$b_i = \frac{g^{r_i} \prod_{j=1}^{n} g_j^{A_{ji}}}{g'_i}, i = 1, ..., n.$$

At this point, the proof in [6] reached the conclusion that $b_i = 1, i = 0, ..., n$. This conclusion is only correct if $b_i \in G_q, i = 0, ..., n$, as shown below.

Suppose $\mathcal{C}$ is the vector space that is spanned by the set $S$ of all vectors

$$u = (1, c_1, c_2, ..., c_n)$$

such that $c_1, c_2, ..., c_n$ satisfy equation (10). As $b_i \in G_q, i = 0, ..., n$, we can assume $b_i = g^{e_i}, i = 0, ..., n$, where $e_i \in Z_q$. Define the vector $e = (e_0, ..., e_n)$, each vector $u \in S$ satisfies the equation

$$eu = 0$$

If $dim(\mathcal{C}) < n+1$, the size of the set $S$ is at most $q^{n-1}$ and so the probability that equation (10) holds is at most $q^{n-1}/q^n = 1/q$, which is negligible. If $dim(\mathcal{C}) = n + 1$, then $e = 0$ and so $b_i = 1, i = 0, ..., n$.

By repeating the same argument for $m'$ and $\{m'_i\}$, Lemma 1 has been proved.

### 4.2 Millimix Attack

**Description.** Millimix is vulnerable to two attacks. An attack similar to the one against Furukawa-Sako01 mix-net described above can be applied to Millimix. That is a malicious mix-server can output incorrect ciphertexts without being detected and the success chance is at least 50%. The attack can be prevented using the same proposed countermeasure. That is, at the beginning of the verification protocol, the verifier must verify if $\alpha'_i, \beta'_i \in G_q$, $i = 1, 2$. This is the same as calculating the Legendre symbol of $\alpha'_i, \beta'_i$, for which the algorithm described in [16] (p. 73) requires one modular multiplication.

In the following, we describe a second attack and a method of countering the attack. As noted earlier, the original $DISPEP$ in [11] computes $(y_{s1}, g_{s1}) = (\alpha_1/\alpha'_1, \beta_1/\beta'_1)$ and $(y_{s2}, g_{s2}) = (\alpha_1/\alpha'_2, \beta_1/\beta'_2)$ as two Schnorr public keys and shows knowledge of one of the Schnorr private keys using Disjunctive Schnorr identification protocol. However, this requires the mix-server to know the El Gamal private key $x$ of the ciphertexts, which is not acceptable. To remove this problem, we show a corrected version of $DISPEP$, which uses the approach in $PEP$, and then show that even with this modification, the correctness of the mix-net can be broken.

**Modified** $DISPEP$**:** $DISPEP$ proves that a ciphertext $(\alpha_1, \beta_1)$ represents a re-encryption of one of the two ciphertexts $(\alpha'_1, \beta'_1)$ and $(\alpha'_2, \beta'_2)$. Compute

$$(y_{s1}, g_{s1}) = ((\alpha_1/\alpha'_1)^{z_1}(\beta_1/\beta'_1), y^{z_1}g)$$
$$(y_{s2}, g_{s2}) = ((\alpha_1/\alpha'_2)^{z_2}(\beta_1/\beta'_2), y^{z_2}g)$$

as two Schnorr public keys. Assume w.l.o.g. that $(\alpha_1, \beta_1)$ is a re-encryption of $(\alpha'_1, \beta'_1)$, then there exists $\gamma_1 \in Z_q$ such that $(y_{s1}, g_{s1}) = ((y^{z_1}g)^{\gamma_1}, y^{z_1}g)$. The prover (mix-server) uses Disjunctive Schnorr identification protocol with $(y_{s1}, g_{s1}), (y_{s2}, g_{s2})$ to show that it knows $\gamma_1$.

**Attack against Verification:**
The attack exploits the fact that the exponents $z$ in $PEP$ and $z_1, z_2$ in $DISPEP$ can be arbitrarily chosen. Let $(\alpha_1, \beta_1)$ and $(\alpha_2, \beta_2)$ denote the two input ciphertexts to a switching gate of a malicious mix-server. The server computes its output ciphertexts as follows.

$$(\alpha_1', \beta_1') = (\alpha_1 y^{-r_1 - s_1 z_1} g^{-s_1}, \beta_1 g^{-r_1})$$
$$(\alpha_2', \beta_2') = (\alpha_2 y^{-r_2 + s_1 z_1 - sz} g^{s_1 - s}, \beta_2 g^{-r_2})$$

Although $(\alpha_1', \beta_1'), (\alpha_2', \beta_2')$ are not valid outputs of the switching gate, but, using $PEP$ and $DISPEP$ the server can still show that: $(i)$ $(\alpha_1'\alpha_2', \beta_1'\beta_2')$ is the re-encryption of $(\alpha_1\alpha_2, \beta_1\beta_2)$, and $(ii)$ either $(\alpha_1', \beta_1')$ or $(\alpha_2', \beta_2')$ re-encrypts $(\alpha_1, \beta_1)$. To show $(i)$, the server computes

$$\begin{aligned}(\alpha/\alpha', \beta/\beta') &= (\alpha_1\alpha_2/\alpha_1'\alpha_2', \beta_1\beta_2/\beta_1'\beta_2') \\ &= (y^{r_1+r_2+sz} g^s, g^{r_1+r_2}) \\ (y_s, g_s) &= ((\alpha/\alpha')^z (\beta/\beta'), y^z g) = ((y^z g)^{r_1+r_2+sz}, y^z g) \\ &= (g_s^{r_1+r_2+sz}, g_s)\end{aligned}$$

Now Schnorr identification protocol will be performed as follows.

1. $\mathcal{P} \longrightarrow \mathcal{V}$: a commitment $w = g_s^e$
2. $\mathcal{P} \longleftarrow \mathcal{V}$: a challenge $c$
3. $\mathcal{P} \longrightarrow \mathcal{V}$: a response $s = e + c(r_1 + r_2 + sz)$

$\mathcal{V}$ then check if $g_s^s = w y_s^c$. This equation is correct and $PEP$ has been broken.

To show $(ii)$, we note that

$$\begin{aligned}(y_{s1}, g_{s1}) &= ((\alpha_1/\alpha_1')^{z_1} (\beta_1/\beta_1'), y^{z_1} g) = ((y^{z_1} g)^{r_1 + s_1 z_1}, y^{z_1} g) \\ &= (g_{s1}^{r_1 + s_1 z_1}, g_{s1})\end{aligned}$$

and so Disjunctive Schnorr identification protocol can be performed as follows.

1. $\mathcal{P} \longrightarrow \mathcal{V}$: two commitments $w_1 = g_{s1}^{e_1}$, $w_2 = g_{s2}^{s_2} y_{s2}^{-c_2}$
2. $\mathcal{P} \longleftarrow \mathcal{V}$: a challenge $c$
3. $\mathcal{P} \longrightarrow \mathcal{V}$: responses $s_1 = e_1 + c_1(r_1 + s_1 z_1)$, $s_2$, $c_1 = c \oplus c_2$, $c_2$

$\mathcal{V}$ then check if $g_{si}^{s_i} = w_i y_{si}^{c_i}$, $i = 1, 2$. These equations hold, so $DISPEP$ succeeds.

**Countermeasures.** To counter the attack against $PEP$, $z$ must be either chosen interactively by the verifier after the switching gate has produced the output, or non-interactively calculated as $z = H(\alpha' \parallel \beta' \parallel \alpha \parallel \beta)$, using a hash function $H : \{0,1\}^* \rightarrow 2^{|q|}$. With this modification, the non-interactive version of the protocol will be as follows.

To prove that $(\alpha', \beta')$ is a re-encryption of $(\alpha, \beta)$, the prover provides a tuple $(z, c, s)$. A verifier can verify the proof by checking if

$$z \stackrel{?}{=} H(\alpha' \parallel \beta' \parallel \alpha \parallel \beta) \bmod q$$
$$c \stackrel{?}{=} H(g' \parallel y' \parallel g'^s y'^c) \bmod q$$

where $(y', g') = ((\alpha/\alpha')^z (\beta/\beta'), y^z g)$.

*DISPEP* can be modified in a similar way. That is both $z_1$ and $z_2$ must be either chosen by the verifier after the switching gate has produced the output, or computed as $z_1 = z_2 = H(\alpha'_1 \parallel \beta'_1 \parallel \alpha'_2 \parallel \beta'_2 \parallel \alpha_1 \parallel \beta_1 \parallel \alpha_2 \parallel \beta_2)$. The prover then performs Disjunctive Schnorr identification (or signature) protocol, in which the public keys are

$$(y_{s1}, g_{s1}) = ((\alpha_1/\alpha'_1)^{z_1}(\beta_1/\beta'_1), y^{z_1}g)$$
$$(y_{s2}, g_{s2}) = ((\alpha_1/\alpha'_2)^{z_2}(\beta_1/\beta'_2), y^{z_2}g)$$

**Security.** Verification protocol in Millimix has been proved to be complete, sound and honest-verifier zero-knowledge. In the following, we show the effect of the above attack on the proof of soundness and note that completeness and zero-knowledgeness proofs will not be affected by the proposed attack.

The proof of soundness is based on Lemmas 2 and 3 taken from [11]. The proof of the Lemmas have not been given in the original paper and in any case because of the attack, the proofs must be revisited. We show a revised statement and proof of Lemma 2, which shows the importance of choosing $z$ carefully. Lemma 3 can be revised similarly.

**Lemma 2.** *Let* $(\alpha, \beta)$ *and* $(\alpha', \beta')$ *be two ciphertexts for which PEP produces* accept *response.*

- *if* $z$ *is chosen by the prover, then* $(\alpha', \beta')$ *is not necessarily a valid re-encryption of* $(\alpha, \beta)$.
- *if* $z$ *is chosen by the verifier or computed by hash function as shown above, then either* $(\alpha', \beta')$ *is a valid re-encryption of* $(\alpha, \beta)$ *or the prover can find the El Gamal private key* $x$.

*Proof.* Suppose $z$ is chosen by the prover. If $(\alpha', \beta')$ and $(\alpha, \beta)$ have the relationship shown in the attack, then *PEP* outputs accept whereas $(\alpha', \beta')$ is not a valid re-encryption of $(\alpha, \beta)$.

Now let $z$ be chosen by the verifier or computed using a secure hash function. Suppose $K$ is the set of all elements $z$ in $Z_q$ such that the prover knows $o \in Z_q$ satisfying $(\alpha/\alpha')^z(\beta/\beta') = (y^z g)^o$. Let $|K|$ be the number of elements in the set $K$. If $z$ is chosen randomly by the verifier or computed by the hash function whose output is uniformly distributed over $Z_q$, the probability that *PEP* outputs *accept* is $|K|/q$. With sufficiently large $q$, we can assume $|K| \geq 3$. Otherwise, $|K|/q$ is negligible and so is the success chance of *PEP*.

So w.l.o.g., assume there exists three distinct elements $z_0$, $z_1$ and $z_2$ in $K$. Let $\alpha/\alpha' = g^u$ and $\beta/\beta' = g^v$. The prover knows $o_0, o_1, o_2 \in Z_q$ satisfying $(\alpha/\alpha')^{z_i}(\beta/\beta') = (y^{z_i}g)^{o_i}$, $i = 0, 1, 2$ and so has the following system of three linear equations with three unknowns $u$, $v$ and $x$:

$$\begin{cases} z_0 u + v - o_0 z_0 x = o_0 \\ z_1 u + v - o_1 z_1 x = o_1 \\ z_2 u + v - o_2 z_2 x = o_2 \end{cases}$$

As $\alpha, \beta, \alpha', \beta' \in G_q$, then $u, v, x$ must exist, and so the system of equations must have a solution. If the solution is unique, the prover will be able to solve it and find the value of $x$ and that demonstrates a knowledge extractor for $x$.

On the other hand, if the system has more than one solution, the following determinants are equal zero.

$$det = \begin{vmatrix} z_0 & 1 & -o_0 z_0 \\ z_1 & 1 & -o_1 z_1 \\ z_2 & 1 & -o_2 z_2 \end{vmatrix} = 0$$

$$det_x = \begin{vmatrix} z_0 & 1 & -o_0 \\ z_1 & 1 & -o_1 \\ z_2 & 1 & -o_2 \end{vmatrix} = 0$$

This implies that,

$$\begin{aligned} 0 &= det + z_0 det_x \\ &= z_0 z_2 o_0 - z_0 z_2 o_2 + z_2 z_1 o_2 - z_2 z_1 o_1 + z_1 z_0 o_1 - z_1 z_0 o_0 \\ &\quad + z_0^2 o_2 - z_0^2 o_1 - z_0 z_1 o_2 + z_0 z_2 o_1 + z_0 z_1 o_0 - z_0 z_2 o_0 \\ &= (o_2 - o_1)(z_0 - z_1)(z_0 - z_2) \end{aligned}$$

and so $o_2 = o_1$. This leads to $u = vx$, which means that $\alpha/\alpha' = (\beta/\beta')^x$ and so $(\alpha', \beta')$ is a valid re-encryption of $(\alpha, \beta)$.

**Lemma 3.** *Let* $(\alpha_1, \beta_1)$, $(\alpha'_1, \beta'_1)$ *and* $(\alpha'_2, \beta'_2)$ *be ciphertexts for which* $DISPEP$ *produces* accept *response.*

- *if* $z_1$ *and* $z_2$ *are chosen by the prover, then* $(\alpha_1, \beta_1)$ *is not necessarily a valid re-encryption of either* $(\alpha'_1, \beta'_1)$ *or* $(\alpha'_2, \beta'_2)$.
- *if* $z_1$ *and* $z_2$ *are chosen by the verifier or computed by hash function as shown above, then either* $(\alpha_1, \beta_1)$ *is a valid re-encryption of either* $(\alpha'_1, \beta'_1)$ *or* $(\alpha'_2, \beta'_2)$ *or the prover can find the El Gamal private key* $x$.

## 5 Conclusion

In this paper, we presented attacks against several universally resilient mix-nets and showed countermeasures against these attacks. We also analyzed security and efficiency of the proposed countermeasures. The first attack that is shown against Furukawa-Sako01 mix-net and Millimix can also be used against a number of other mix-nets, more specifically, in breaking proofs of correctness of the mixing phase in MiP-1, MiP-2 [2,3] and Neff01 [19], and breaking proofs of correctness of the decryption phase in Abe98 [1] and MiP-2 [2,3]. MiP-1 and MiP-2 are very similar to Millimix and so the first attack can be similarly used. The correctness of Neff01 mix-net relies on the Iterated Logarithm Multiplication Proof protocol that can be easily subjected to the first attack. Abe98 protocol uses threshold El Gamal Decryption and introduces a way of jointly decrypting

and proving correctness. It can be shown that the attack is also applicable in this case. The details of these attacks will be provided in the final version of this paper. We note that all these proofs are based on the hardness of discrete logarithm problem. It is conceivable that the attack could have wider implications for a range of proofs that are based on discrete logarithm assumption and so must be carefully considered in all such proofs. The second attack breaks the verification protocol of Millimix. The attack can be countered by carefully choosing the challenge.

## References


1. M. Abe. Universally verifiable mix-net with verification work independent of the number of mix-servers. In K. Nyberg, editor, EUROCRYPT '98, pages 437-447. Springer-Verlag, 1998. LNCS No. 1403.
2. M. Abe. A mix-network on permutation networks. In K.Y. Lam, C. Xing, and E. Okamoto, editors, ASIACRYPT '99, pages 258-273, 1999. LNCS no. 1716.
3. M. Abe and F. Hoshino. Remarks on Mix-Network Based on Permutation Networks. Public Key Cryptography 2001, pages 317-324.
4. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM, 24(2):84-88, 1981.
5. Y. Desmedt and K. Kurosawa. How to break a practical mix and design a new one. In B. Preneel, editor, EUROCRYPT '00, pages 557-572. Springer-Verlag, 2000. LNCS no. 1807.
6. J. Furukawa and K. Sako. An Efficient Scheme for Proving a Shuffle, pages 368 ff. J. Kilian (Ed.), CRYPTO 2001. LNCS 2139
7. E. Gabber, P. Gibbons, Y. Matias, and A. Mayer. How to make personalized Web browsing simple, secure, and anonymous. In R. Hirschfeld, editor, Financial Cryptography '97, pages 17-31, 1997.
8. M. Jakobsson. A practical mix. In K. Nyberg, editor, EUROCRYPT '98, pages 448-461. Springer-Verlag,1998. LNCS No. 1403.
9. M. Jakobsson and D. M'Raihi. Mix-based electronic payments. In E. Tavares S and H.Meijer, editors, SAC '93, pages 057-473. Springer-Verlag, 1998. LNCS no. 1505.
10. M. Jakobsson. Flash mixing. In PODC '99, pages 83-89. ACM, 1999.
11. M. Jakobsson and A. Juels. Millimix: Mixing in small batches, 1999. DIMACS Technical Report 99-33.
12. M. Jakobsson and A. Juels. Mix and match: Secure function evaluation via ciphertexts. In T. Okamoto, editor, ASIACRYPT '00, pages 162-177, 2000. LNCS No. 1976.
13. M. Jakobsson, A. Juels, "An Optimally Robust Hybrid Mix Network", PODC '01
14. M. Jakobsson, A. Juels and R. Rivest, "Making Mix Nets Robust For Electronic Voting By Randomized Partial Checking", USENIX Security '02.
15. A. Juels. Targeted advertising and privacy too. In D. Naccache, editor, RSA Conference Cryptographers' Track, 2801.
16. A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. Handbook of Applied Cryptography. CRC Press 1996.
17. Markus Michels, Patrick Horster: Some Remarks on a Receipt-Free and Universally Verifiable Mix-Type Voting Scheme. ASIACRYPT 1996, pages 125-132.
18. M. Mitomo and K. Kurosawa. Attack for flash mix. In T. Okamoto, editor, ASIACRYPT '00, pages 192-204, 2000. LNCS No. 1976.

19. A. Neff, A verifiable secret shuffle and its application to e-voting. In P. Samarati, editor, ACM CCS '01, pages 116-125. ACM Press, 2001.
20. W. Ogata, K. Kurosawa, K. Sako, and K. Takatani. Fault tolerant anonymous channel. In Proc. ICICS '97, pages 440-444, 1997. LNCS No. 1334.
21. M. Ohkubo and M. Abe. A length-invariant hybrid mix. In T. Okamoto, editor, ASIACRYPT '00, pages 178-191, 2000. LNCS No. 1976.
22. C. Park, K. Itoh, and K. Kurosawa. Efficient anonymous channel and all/nothing election scheme. In T. Helleseth, editor, EUROCRYPT '93, pages 248-259. Springer-Verlag, 1993. LNCS No. 765.
23. B. Pfitzmann. Breaking an Efficient Anonymous Channel. EUROCRYPT '94, pages 332-340. Springer-Verlag, 1995. LNCS No. 950.
24. K. Sako and J. Kilian. Receipt-free mix-type voting scheme - a practical solution to the implementation of a voting booth. In L.C. Guillou and J.-J. Quisquater, editors, EUROCRYPT '95. Springer-Verlag, 1995. LNCS No. 921.
25. A. Waksman. A permutation network. J. ACM, 15(1):159-163, 1968.

# Improving Onion Notation

Richard Clayton

University of Cambridge, Computer Laboratory, William Gates Building,
15 JJ Thompson Avenue, Cambridge CB3 0FD, United Kingdom
richard.clayton@cl.cam.ac.uk

**Abstract.** Several different notations are used in the literature of MIX networks to describe the nested encrypted structures now widely known as "onions". The shortcomings of these notations are described and a new notation is proposed, that as well as having some advantages from a typographical point of view, is also far clearer to read and to reason about. The proposed notation generated a lively debate at the PET2003 workshop and the various views, and alternative proposals, are reported upon.The workshop participants did not reach any consensus on improving onion notation, but there is now a heightened awareness of the problems that can arise with existing representations.

## 1 Introduction

Chaum's original paper on MIX systems [5] used a functional notation for expressing encryption. For example, a user's message $T$ which is to be delivered to the destination $A$ would be encrypted using $A$'s public key $K_a$ and the address A would be appended. This would all then be encrypted (the original paper called this "sealed") with the public key $K_1$ of MIX machine $M_1$. With the addition of random nonces $R_0$ and $R_1$ this would be expressed as a message for $M_1$:

$$K_1(R_1, K_a(R_0, T), A)$$

This message (along with the destination address $M_1$ and another nonce $R_2$) could be further encrypted with the public key $K_2$ of MIX $M_2$ and so on to involve as many MIX systems as might be required.

These nested structures are now widely known as "onions", a term first brought to prominence by Goldschlag et al [9]. A related notion of layered encryption was called a "telescope" by Zero Knowledge [8].

Nested encryption has been widely discussed, but Chaum's original notation is not always used, perhaps because it was so closely associated with the use of public key encryption, rather than a generic sealing operation. Other authors, such as Ohkubo and Abe [11] use an explicit notation for the encryption function, such as $\mathcal{E}$. In this notation, encryption with a key $K_a$ becomes $\mathcal{E}_{K_a}$ and our example becomes:

$$\mathcal{E}_{K_1}(R_1, \mathcal{E}_{K_a}(R_0, T), A)$$

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 81–87, 2003.

Another group of authors, such as Serjantov [13], have used a bracketed subscript notation, similar to BAN logic [3] and later used in the Spi calculus [1]. In this notation the encryption of message $T$ under key $K$ is expressed as $\{T\}_K$. Therefore the onion example becomes:

$$\{R_1, \{R_0, T\}_{K_a}, A\}_{K_1}$$

A plethora of different notations is perhaps inevitable because authors use the ones with which they are familiar. However, it is possible to set out some objective criteria for assessing notation. Designing a new notation that better meets these criteria does of course add to the existing babel, but if it is sufficient improvement then this may be acceptable.

## 2 Assessing Notations

The history of mathematical notation [4] is littered with notations that have failed to gain wide acceptance, or that have been swept away by later, simpler, schemes. In some cases such as Frege's *Begriffsschrift* [6] they have actively prevented contemporaries from understanding the scientific contribution. [10] A good notation will be simple to read, write and typeset. In particular:

**It Will Fit onto a Single Line:** Notation that uses vertical layout to convey information has seldom survived when alternatives have been developed that have the same height as normal text. Frege's notation was especially weak in this respect.

**It Will Be Easy to Write:** Onions are regularly written on whiteboards, in typographical systems such as LaTeX and in presentation systems such as PowerPoint. The notation should produce similar looking results no matter what medium is used. In particular, specification documents such as the RFCs produced by the Internet Engineering Task Force (IETF) are constrained to use ASCII characters and this is extremely restrictive.

**It Will Be Easy to Read:** Familiarity will make even the most exotic notation appear normal. However, practical issues relating to poor eyesight must be kept in mind, and multiple levels of subscript or superscript should be avoided because of the size reduction that is conventionally applied.

**It Will Be Easy to Comprehend:** The use of deeply nested brackets needs to be avoided. If the levels of nesting need to be joined up in pencil before they can be understood, then the notation is preventing understanding rather than enhancing it.

**It Will Allow Simple Generalisation:** All of the notations currently in use express onions of arbitrary complexity by means of ellipsis (...) notation. However, where more than one ellipsis is needed then it can become unclear how deeply nested constructs are built. For example:

$$K_n(R_n, K_{n-1}(...K_2(R_2, K_1(R_1, K_a(R_0, T), A), M_1), ...), M_{n-1})$$

may present some difficulties in understanding exactly how to match up the left and right hand ends of the expression.

**It Will Allow Errors to Be Easily Detected:** Related parts of the onion should be presented together whenever possible. If keys and addresses are at opposite ends of an expression then it is hard to spot a mistake such as:

$$K_3(R_3, K_2(R_2, K_1(R_1, K_a(R_0, T), A), M_3), M_2)$$

where the address $M_3$ should be $M_1$.

**Etcetera:** It is possible to go on to identify many other issues worthy of consideration. For example, the Cognitive Dimensions of Notations framework [2] lists a total of 13 dimensions that should be considered, especially when the whole life-cycle of the notation is considered and not just its appearance in academic papers and conference presentations.

## 3 Proposed Onion Notation

Clearly, from the list of requirements, an improved notation for onions will fit on a single line, will avoid multiple levels of subscripts, and will not nest brackets. An entirely straightforward scheme for achieving this would be to write down the onion in the temporal order in which it is constructed:

$$|R_0, T\#K_a|R_1, A\#K_1$$

where one reads | as starting a section and # as meaning "use the following key to encode the preceding section".

This notation may be adequate for theoreticians who are seldom concerned about the exact order of the concatenated components within a particular onion layer. However to satisfy those who consider this ordering to be important we need to add a symbol to express "the result of the last encryption operation". Here, in a typeset paper the LaTeX `\star` symbol $\star$ is used for this purpose, though substituting an asterisk ($*$) would not cause confusion in other contexts. Our onion expression for the packet sent to MIX $M_1$ now becomes:

$$|R_0, T\#K_a|R_1, \star, A\#K_1$$

which we should read as being a packet encrypted under the public key $K_1$ that contains the nonce $R_1$, then the packet encrypted under the key $K_a$ and then the address of $A$; where the packet for $A$ contains the nonce $R_0$ and the user's message $T$.

### 3.1 Advantages of the ⋆ Notation

Some advantages of this notation will be immediately apparent. One can generalise it to an onion of arbitrary complexity without any need to count brackets, match up MIX identifiers with keys or any of the other tedious proof-reading that traditional notations require. For example it is far easier to construct and check the generalised onion:

$$|R_0, T\#K_a|R_1, \star, A\#K_1|R_2, \star, M_1\#K_2|...\#K_{n-1}|R_n, \star, M_{n-1}\#K_n$$

than the traditional:

$$K_n(R_n, K_{n-1}(...K_2(R_2, K_1(R_1, K_a(R_0, T), A), M_1), ...), M_{n-1})$$

The small advantage this gives the author pales into insignificance when compared to the significant advantage that it gives to every future reader.

A more subtle advantage is that it is now possible to reason about the various MIXs directly from the notation. Assuming reasonable security precautions have been taken, only MIX $M_i$ will be able to decrypt material encoded under its public key $K_i$. Hence it is immediately apparent that it will only have access to that part of the onion which lies in the section to the left of any $\#K_i$ within the onion. In the running example this will mean that it can only "see" the value $R_i, \star, M_{i-1}$ and from that, it will be possible to derive (informal) correctness results.

### 3.2 A Further Complication

One further piece of notation is needed, to cover onions that are encrypted to more than one key per MIX, such as in Pfitzmann & Waidner's 1986 proposal [12] for avoiding end-to-end retransmissions in MIX networks. Using the traditional notation of the original paper (but labelling consistently with this one), the encrypted message $X_i$ that is delivered to MIX $M_i$ is recursively defined as:

$$\begin{aligned} X_a &: K_a(T) \\ X_n &: K_n(k_n, A), k_n(X_a) \\ X_i &: K_i(k_i, M_{i+1}, k_{i+1}, M_{i+2}), k_i(X_{i+1}) \end{aligned}$$

where key $k_i$ is a second key known to MIX $M_i$ in addition to the usual $K_i$ value.

To handle this case, the new notation needs one further piece of syntax. The $\star$ values must be enumerated to indicate which of the preceding encrypted results is to be used. We define $\star_0$ to be the same as $\star$, i.e. the result of the immediately preceding encryption operation. $\star_1$ is then defined to be the result of the encryption operation before that (etc as needed). With this extension the scheme becomes:

$$\begin{aligned} X_a &: |T\#K_a \\ X_n &: |X_a\#k_n|k_n, A\#K_n| \star_0 \star_1 \\ X_i &: |X_{i+1}\#k_i|k_i, M_{i+1}, k_{i+1}, M_{i+2}\#K_i| \star_0 \star_1 \end{aligned}$$

Expanding the onion by even just a single level demonstrates the advantage of the new notation. $X_i$ could also have been expressed as:

$$K_i(k_i, M_{i+1}, k_{i+1}, M_{i+2}), k_i(K_{i+1}(k_{i+1}, M_{i+2}, k_{i+2}, M_{i+3}), k_{i+1}(X_{i+2}))$$

whereas in the new notation it becomes:

$$|X_{i+2}\#k_{i+1}|k_{i+1}, M_{i+2}, k_{i+2}, M_{i+3}\#K_{i+1}|\star_0\star_1\#k_i|k_i, M_{i+1}, k_{i+1}, M_{i+2}\#K_i|\star_0\star_1$$

which is rather simpler to parse. In fact, it is even quite straightforward to avoid the recursive definition entirely, which although elegant, takes a little while to understand. Using the $\star$ notation the full onion can be written out without the risk of alienating the reader, something the original authors wisely eschewed:

$$\begin{aligned}
&| T\#k_a \\
&| k_n, A\#K_n| \star_0 \star_1\#k_{n-1} \\
&| k_{n-1}, M_n, k_n, A\#K_{n-1}| \star_0 \star_1\#k_{n-2} \\
&| k_{n-2}, M_{n-1}, k_{n-1}, M_n\#K_{n-2}| \star_0 \star_1\#k_{n-3} \\
&| k_{n-3}, M_{n-2}, k_{n-2}, M_{n-1}\#K_{n-3}| \star_0 \star_1\#k_{n-4} \\
&| k_{n-4}, M_{n-3}, k_{n-3}, M_{n-2}\#K_{n-4}| \star_0 \star_1\#k_{n-5}...
\end{aligned}$$

From this we can, by simple inspection, directly reason which other keys the possessor of $K_i$ will have access to (i.e.: $k_i$ and $k_{i+1}$). However, possession of these keys will only allow access to material encrypted under $K_{i+1}$, $K_{i+2}$ or $K_{i+3}$, and hence the system security properties hold.

With the improved notation, it will still be possible to do this type of analysis even with the presence of otherwise distracting details. These would include the use of nonces to prevent output message identification and the addition of padding to prevent traffic analysis attacks based on the size of message headers.

## 4 Workshop Discussion

The presentation of the paper at the workshop was followed by a short discussion:

**Adam Shostack:** On your presentation slides you showed how the brackets could be linked together by different levels of underlining. I think that was very visual and appropriate usage of underlining and overlining would be an elegant way to portray exactly which elements are encrypted with particular keys.

**Sandra Steinbrecher:** Your notation seems to be created to assist implementers, but many researchers will be coming from a mathematical background and will cope with the existing notations.

**Rachel Greenstadt:** It's important to be able to come from related fields such as crypto or security and read our papers. Everyone will be used to brackets and so your notation must have brackets.

**Richard Clayton:** It is possible to dispose of the # notation and use brackets for the action of the encryption keys. The $\star$ notation remains the same so as to avoid all the nesting. i.e.:

$$K_a(R_0, T)|K_1(R_1, \star, A)|K_2(R_2, \star, M_1)$$

**James Alexander:** Perhaps language theory has something to contribute. In that field the representation and the notation of the underlying model will differ. There should be no problem with notations for different purposes if there is a clear mapping between them, perhaps using a functional notation.

**Dogan Kesdogan:** Adding yet another notation might not be entirely helpful.

**Christian Grothoff:** I'd suggest that a functional notation might meet the need for clarity without the hardships caused by the introduction of new syntactic elements. For example, defining a function and then using a standard notation for composition would look like this:

| | |
|---|---|
| Definition: | $\mathcal{E}(a, b)(x) := K_a(R_a, x, b)$ |
| Encryption: | $\big(\mathcal{E}(M_3, M_2) \circ \mathcal{E}(M_2, M_1) \circ \mathcal{E}(M_1, A)\big)\ (T)$ |

**Andrei Serjantov:** I made a specific point of using the notation from the community in which my papers were originally introduced.

**Paul Syverson:** I think the key point is that there is not just one community working on onions. People are coming at them from theory, from PETs and from the crypto community. Your notation may assist the people who are writing code, but it may not be very helpful to those who just want to analyse the validity of their schemes, so I am not sure that there is as big a win here as you are suggesting.

## 5 Conclusions

A notation for expressing the construction of onions has been presented that is as concise as existing notations. By mirroring the temporal construction of the onions the reader avoids having to count brackets backwards and forwards to establish the author's intent. The notation is also capable of dealing with complex cases with encryption with multiple keys.

However, the notation was not widely welcomed by the workshop, although people were prepared to accept that existing papers sometimes manage to obscure the mechanisms they present by poor use of notation. There seems to be a role for programme committees in ensuring some consistency of notation for accepted papers to prevent unnecessary divergence from established practice.

Finally, when pondering the utility of change, one might note the observation of Leibniz [7] (who was one of the most successful innovators of mathematical notation, several of his inventions having survived to the present day):

> *"In signs one observes an advantage in discovery which is greatest when they express the exact nature of a thing briefly and, as it were, picture it; then indeed the labour of thought is wonderfully diminished."*

## Acknowledgements

I would like to thank George Danezis and Andrei Serjantov for their initial help in ensuring I did not produce an unacceptably obscure notation. George also took notes of the workshop discussion. I'd also like to thank the anonymous referees for drawing my attention to Frege's notation-induced failure and for making me face up to the significant problems that brackets were causing, so that I finally stumbled upon the simplicity that "|" could provide.

## References

1. M. Abadi and A. D. Gordon: A calculus for cryptographic protocols: The spi calculus. Information and Computation, 148(1), 1999, pp. 1–70.
2. A. Blackwell and T. Green: Notational Systems - the Cognitive Dimensions of Notations framework. In J. M. Carroll (Ed.): HCI Models, Theories, and Frameworks: Towards a Multidisciplinary Science. Morgan Kaufmann Publishers, 2003.
3. M. Burrows, M. Abadi and R. Needham: A Logic of Authentication. ACM Trans. on Computer Systems 8(1), 1990, pp. 18–36.
4. F. Cajori: A history of mathematical notations. The Open Court Publishing Company, Chicago Il. 1928–9.
5. D. Chaum: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Comm. ACM 24(2), 1981, pp. 84–88.
6. G. Frege: Begriffsschrift. Eine der arithmetischen nachgebildeten Formelsprache des reinen Denkens. Halle, 1879.
7. C. I. Gerhardt: Briefwechsel von G.W.Liebniz mit Mathematikern, Vol I. quoted in para. #540 of [4].
8. I. Goldberg and A. Shostack: Freedom Network 1.0 Architecture, October 10 2001. `http://www.homeport.org/~adam/zeroknowledgewhitepapers/archnotech.pdf`
9. D. M. Goldschlag, M. G. Reed, and P. F. Syverson: Hiding Routing Information. In R. Anderson (Ed.): Information Hiding, LNCS 1174, Springer Verlag 1996, pp. 137–150.
10. R. Mendelsohn: Begriffsschrift in Modern Notation. `http://comet.lehman.cuny.edu/mendel/papers/Adobe%20Versions/AdobeNewBGForms.pdf`
11. M. Ohkubo and M. Abe: A Length-Invariant Hybrid Mix. In T. Okamoto (Ed.): ASIACRYPT 2000, LNCS 1976, Springer Verlag 2000, pp. 178–191.
12. A. Pfitzmann and M. Waidner: Networks without User Observability. Computer Security 6(2), 1987, pp. 158-166.
13. A. Serjantov: Anonymizing censorship Resistant Systems In P.Druschel et al (Ed.): First International Workshop, IPTPS 2002, Cambridge Ma. USA, March 2002, LNCS 2429, Springer Verlag 2002, pp. 111-120.

# Engineering Privacy in Public: Confounding Face Recognition

James Alexander and Jonathan Smith

Department of Computer and Information Science, University of Pennsylvania
200 South 33rd Street
Philadelphia, PA 19104-6389
{jalex,jms}@cis.upenn.edu

**Abstract.** The objective of DARPA's Human ID at a Distance (HID) program "is to develop automated biometric identification technologies to detect, recognize and identify humans at great distances." While nominally intended for security applications, if deployed widely, such technologies could become an enormous privacy threat, making practical the automatic surveillance of individuals on a grand scale. Face recognition, as the HID technology most rapidly approaching maturity, deserves immediate research attention in order to understand its strengths and limitations, with an objective of reliably foiling it when it is used inappropriately. This paper is a status report for a research program designed to achieve this objective within a larger goal of similarly defeating all HID technologies.

## 1 Introduction

Surveillance cameras, as with most information technologies, continually improve in performance while simultaneously decreasing in cost. This falling cost-of-entry for quality imaging has led to widespread deployment of cameras in public and semi-public areas, often with no public oversight of how these cameras are used. The possibility of pooling many lenses together into a larger sensor network is disturbing enough, allowing the possibility of tracking an individual's movements wherever these lenses can see. In the past, monitoring such a network to follow everyone in view would have entailed prohibitive personnel costs. However, pairing together such a network with an identification-at-a-distance technology such as automatic face recognition, could greatly reduce or eliminate this final cost barrier. Although not yet a mature technology, we anticipate that face recognition will continue to advance with the other information technologies. Given the recent great increase in interest in leading-edge security technologies, conditions are good for rapid performance improvements, so the emergence of face recognition as a low-cost commodity may not be far off. In this paper, we begin an investigation into how current face recognition techniques might be defeated, both to explore their limitations in legitimate security applications, as well as preparing for the possibility of future misuses.

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 88–106, 2003.

The chief contribution of this paper is the empirical results of testing various face recognition countermeasures against the first of several systems, as well as a framework for interpreting those results. In the next section, we provide the details of our experimental conditions and methods. Section 3 presents our preliminary findings: what did and did not work, the patterns we see emerging, and the beginnings of a framework for modeling those patterns. Section 4 offers some thoughts on measuring privacy problems in general, which is the ultimate goal of the research program that the current experiment has initiated. In Sect. 5 we briefly discuss some related efforts, and finally we conclude with some remarks on how we intend to proceed in the future.

## 2 Description of Experiment

The face recognition system used in this experiment is an implementation of the eigenfaces method [36], incorporating the optimizations that Moon and Phillips [17] recommend in order to achieve the best performance among eigenfaces systems. Specifically, it is a system that treats training images as vectors, stacks them into a matrix, and estimates eigenvectors for that matrix using principal component analysis (PCA). It then projects probe[1] images into the "face space" for which the eigenvectors form a basis, and uses a distance measure to find the closest matches to the projections of the training images into the same space. The particular distance metric used is the "Mahalanobis angle," a novel metric proposed in [17] that performs the best overall among the seven such similarity measures tested in Moon and Phillips's empirical evaluation. The measure simply takes the venerable "cosine" similarity measure and scales each axis in face space according to the actual variance in the data along that axis. For reference, it is defined as shown in (1), where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the projections of the images being compared in face space and $z_i = 1/\lambda_i^{1/2}$, and $\lambda_i$ is the eigenvalue corresponding to the $i$th eigenvector.

$$d_{\mathrm{MA}}(\boldsymbol{x}, \boldsymbol{y}) = -\frac{\sum_{i=1}^{k} x_i y_i z_i}{\left[\sum_{i=1}^{k} (x_i)^2 \sum_{i=1}^{k} (y_i)^2\right]^{1/2}} \tag{1}$$

The system uses these distances to produce a sorted list of possible matches in the training set, ideally in order of decreasing likelihood. Among other minor optimizations, this system also incorporates some simple image filtering to lessen sensitivity to varying light levels that might exist across its input image sets.

This particular eigenfaces implementation was actually used as a baseline in the FERET [24,19] evaluation, a government study of the state of face recognition technology that ran from 1993 to 1997. An extremely valuable contribution that FERET made to the field of face recognition was its corpus: 14051 256x384

[1] The face recognition literature standardly uses the term *probe* for images to be identified, i.e. members of testing data sets, and uses *gallery* for the images that the system will learn face data from, i.e. training data.

eight-bit grayscale facial images. We used the subset of these images, 3816 images in total, that had been annotated for eye, nose and mouth position. The eigenfaces implementation we tested requires the position of the eyes as one of its inputs since it lacks the ability to locate the eyes unaided. The system uses these coordinates in order to scale the images to a standard size and to crop them with an elliptical template, removing irrelevant background information. Note that supplying this information to the system only makes our countermeasure task harder: in developing our techniques to fool the system, we had to ignore the possibility of preventing it from locating the eyes in the first place, which could prove to be the Achilles heel of more sophisticated systems.

We randomly divided the 3816 images into two sets, 2/3 training and 1/3 testing, in order to establish a base performance, measured as the fraction of answers which had at least one correct match in the top ten. The result (about 75%) was consistent with that reported by Moon and Phillips. This initial run was only needed to confirm that when we added new images at the next stage, we would not unexpectedly degrade the aggregate performance of the system.

Since the performance of the system employed in this first experiment is not state-of-the-art, we endeavored to give the system the best chance possible to get the right answer. Accordingly, we only used frontal images in the testing set and tried to keep lighting conditions reasonably controlled. We also planned to give the system several training examples to work with since, while the system only achieved 75% accuracy when many subjects had only a single training images (and some none at all), it achieves a much more respectable 91% accuracy when considering only subjects with 2 or more training images.

Since we wanted to generate images of our facial recognition countermeasures in real-world use, we couldn't just retouch images in the FERET database, and none of the subjects already in the corpus were available to us for re-imaging. Accordingly, we did not directly use any of the FERET images to test our disguise techniques, but rather we employed them only as distractors. Instead, we took a set of images of two new subjects with a 3 megapixel digital camera, selected 12 images of each subject, modifying them only to match the size and color depth of the FERET images. The set of images selected had small variations in pose and facial expression comparable to those subjects with 12 or more images in the corpus. After adding these new images, a random split of the augmented corpus was made and the experiment was repeated. Aggregate performance decreased slightly (74%), but the system successfully identified *all* of the probe images taken of the new subjects. We expected this high level of performance on these images because of the much larger than average training set (9 images each), and also because it was impossible to duplicate the lighting and equipment used to take the FERET images, so the system could use any such variations to differentiate our images from the distractors.

Next, we randomly split the this augmented corpus once more, and kept this split fixed for all further experiments described here. More precisely, the training set was kept fixed, but many images were added to the probe set: these images were of one of the new subjects employing various candidate countermeasures.

**Fig. 1.** Some sample probes: undisguised, masked with nylon hose, and employing a laser

In generating these images, we made an attempt to control for lighting, pose and facial expressions, with varying degrees of success, in order compare the performance of pairs of countermeasures as accurately as possible. That is, we were interested in measuring the effect of the noise we were deliberately injecting into the system, as opposed to unrelated and unintentional noise. Over 40 images, modified only for size and color depth as before, were tested. Additionally, over 300 more were generated by a morphing algorithm (more on this in Subsect. 3.2) plus a few with minor digital image edits (which are explained in Subsect. 3.4). Figure 1 depicts some examples of our new probes: one of the baseline images as well as a couple of the more successful countermeasures.

Finally, a follow-up experiment was done making use of the AR database [14], which contains a sequence of 26 images of each of 126 people. These image sequences consist of two sessions of 13 images each, varying lighting conditions and facial expression, plus several images with facial occlusions employing sunglasses and a winter scarf. We trained on 10 of the 12 images in each sequence where the facial expression was essentially neutral, reserving 2 of the neutral images for baseline testing, as well as testing the occluded images. These results of these tests lend some statistical weight to our findings, which are presented in the next section.

## 3 Results and Analysis

### 3.1 Scoring Disguises

To analyze the output of the system, we need to model how a face recognition system is likely to be used. We assume a powerful adversary, deploying a face recognition on a large scale using high-quality imaging equipment, and that they might have more than one reference image of each individual in their training data. Based on descriptions of fielded commercial systems, we further assume the system will return as its output a reasonably small number of candidate

matches, which a system operator is expected to verify, either live or offline. Let $N$ be the maximum number of images we think an operator could quickly and accurately verify for a face recognition system deployed on some large scale, i.e. a scale in which a large volume of faces per unit time is expected to be processed. Our goal in designing countermeasures is to keep as many correct identifications out of the top $N$ slots as we can, and we want any correct matches to rank as far down the list as possible. Our evaluation metric, then, should weight those matches that appear earlier over those that appear later. For simplicity, we chose the most obvious decreasing linear function to model this, given as (3.1).

$$w_x(i) = \begin{cases} N - i + 1 & \text{if the candidate in the } i\text{th position} \\ & \text{really is } x \text{ (i.e. a match)} \\ 0 & \text{otherwise} \end{cases}$$

We then sum $w_x$ over $N$ and, for ease of interpretation, we normalize the sum to lie in the interval $[0, 1]$, giving us our score function, shown in (3.1). For concreteness, we chose to use $N = 10$ for the numbers reported here, but this choice is arbitrary and is tunable in our analysis software.

A good countermeasure, then, will score 0 or close to it: 0 represents no correct guesses in the top ten slots. A particularly ineffective disguise might score, for example, 0.6545, which represents correct guesses in positions 1, 2, 3, 6 and 7.

$$\text{score}(x) = \frac{\sum_{i=1}^{N} w_x(i)}{\sum_{i=1}^{N} i}$$

All commercial face recognition systems we are aware of also contain some tunable cut-off for their similarity measure so that the system does not display matches that fall below some minimum likelihood in their model. This allows an arbitrary increase in match accuracy in return for a corresponding decrease in recall, and hence an almost certain increase in the rate of false negatives. However, since we cannot tell where an adversary might set this parameter, we assume, for now, that that this threshold is set so that all top matches are taken seriously. This assumption leads to some difficulties that we will revisit in Subsect. 3.3.

In order to give the reader a frame of reference in which to interpret the "goodness" of any raw scores mentioned later in the text, a few performance statistics on our probe set under this score function might be useful. The whole probe set, including the FERET-supplied distractors, scored an average of 0.2482 with a standard deviation of 0.2365. Note that these statistics indicate that, in the average case, a probe has more than one correct answer in the top ten, and even at one standard deviation below the average, the system still scores better than one correct answer in the tenth position. The median score, 0.1818, corresponds to a correct answer in the first position.

As a baseline for comparison, we took several images of the test subject with a completely undisguised face, and as expected, these images were easily identified, all but one scoring about 0.5 (which is near the one standard deviation point above the mean). The leftmost image in Fig. 1 is representative of these baselines.

**Table 1.** AR test results

| Image Group | Accuracy | Mean Score | StdDev of Score |
|---|---|---|---|
| baseline (5, 18) | 254/255 = 99.6% | 0.6947 | 0.2330 |
| sunglasses (8, 9, 10, 21, 22, 23) | 115/765 = 15.0% | 0.0344 | 0.1125 |
| scarf (11, 12, 13, 24, 25, 26) | 449/765 = 58.7% | 0.2323 | 0.2751 |
| all AR probes | 818/1785 = 45.8% | 0.2136 | 0.3042 |

For reasons of space, we cannot reproduce here all of the images that scored the optimal zero under our score function. The center and right images in Fig. 1, however, are good examples of such images: in one the head is covered with dark nylon hose, whereas the right image results from shining a laser pointer into the camera lens. It is surprisingly easy to do hit the camera lens reliably provided the position of the camera is known, which admittedly is not a particularly realistic assumption, and it is nothing approaching subtle, not to mention potentially hazardous to bystanders. A bright flashlight appears to work just as well without being nearly as dangerous, but otherwise suffers from the same practical drawbacks. Simpler disguises that also worked very well were a pair of mirrored sunglasses, a scarf wrapped around the lower face as if for cold weather, a bandage wrapped around one eye as if the subject had suffered an eye injury, and wide, dark stripes painted onto the face with stage make-up, a technique we will explore more thoroughly later. Other techniques that worked well, but did not quite score zero, included less opaque dark glasses and a lighter-colored pair of nylon hose.

Our tests employing the AR database were consistent with those of our test subject when employing similar countermeasures. These results are summarized in Tab. 1; the AR sequence numbers of the images that comprise each class are provided for reference. The system was able to identify the baseline images with a very high degree of accuracy, but stumbles quite badly when the same faces are disguised with dark glasses, and not quite so badly when the lower face is covered with a scarf. This is consistent with results that have been reported elsewhere [10], suggesting that the eye area contains more discriminatory information than the mouth area.

A very interesting fact arises from examining the disguised AR subjects that were recognized, i.e. those with a score greater than zero: a majority of them score better than one match in the top ten. Moreover, most of these subjects also had more than three on their disguised images identified. A few unlucky souls were identified in all or nearly all of their disguised pictures! Clearly it is not acceptable to just inform such individuals that they are out of luck, so this certainly bears further investigation. Subsection 3.4 discusses some preliminary observations.

Most of the attempts at disguise that did not succeed were not at all surprising: we tried several normal eyeglass frames with clear lenses, and these all performed comparably to the baseline images. This result is consistent with results reported in trials of commercial face recognition systems [20]. One result that was surprising at first, until a pattern began to emerge in the successful

trials, was that a latex nose appliance that had been painted to match the skin of the subject also performed similar to the baseline, even though the apparent shape of the nose was considerably altered. Another try involved use of reflective make-ups in order to use the flash of our camera against it, dazzling it much as we had successfully done with the laser and flashlight. Unfortunately, not enough light was reflected back into the lens to have an effect, which was just as well since it is not clear that a covert face recognition system could employ a flash in any case.

A final interesting failure is an attempt to build a camera-dazzling device using infrared light-emitting diodes. The idea would be to reproduce the effect of the flashlight and laser in a way that would not be visible to an unaided observer. Unfortunately, while our cameras are sensitive to infrared wavelengths, they are not sensitive enough for this purpose: the result was barely visible under ambient light, even with flashes disabled. Based on the apparent brightness of the LEDs in our test photographs, we estimate that the light source would need to be 20 to 100 times brighter in order to have the desired effect. While there are inexpensive infrared LED lasers available that could achieve this, such an illumination level is well into the hazardous range in terms of potential damage to human eyes, particularly since the eye cannot reflexively protect itself from wavelengths it cannot detect in the first place.

We should note that none of the countermeasures we employed to truly devastating effect on the performance of the system under study are undetectable when the system is being monitored by a live observer. This means that in a supervised security application, our disguises would likely prove ineffective: a guard could ask an individual to remove dark glasses, for instance. But in a situation where an individual is being surveilled without his or her knowledge, in a public place or perhaps inside a store, it might be totally reasonable to be wearing moderately tinted lenses or to be bundled up for a cold winter day, without attracting undue attention.

### 3.2 Squeezing More from the Score

Treating all zero-score disguises as equivalent is somewhat unsatisfactory: some actually have correct matches in the teens, while others do not have a correct match until well after the hundredth position. It is not clear, however, that extending the score function to an arbitrarily large $N$ (greater than 10) would be a viable approach: at some point the distance measure must lose statistical significance. Identifying this point precisely is not easy, however; the next subsection will have more to say about this. In this subsection we describe a different approach that has some nice advantages: we chose an arbitrary score threshold of 0.0909[2]. We also chose one of the baseline images[3] to use as a reference image,

[2] This is arbitrary in the sense that we could have chosen any fixed threshold to investigate. This particular threshold corresponds to a single correct identification in the sixth position - one or more matches in the first through fifth position would result in a higher score than this.

[3] ... the one that happened to have the average score of all the baselines. This choice is also arbitrary.

and morphed each of the images that scored less than the threshold into this reference image. A fragment of such a morph is shown in Fig. 2, which should be enough to give a good idea of what results from this process; the full sequence contains more intermediate images.

**Fig. 2.** A sample morph

We can now feed these synthetic images back into the recognition system to see how they fare. The results should give us a better idea of how much of a disguise is need to meet any specific threshold: by counting the number of frames it takes each zero-scoring disguise to reach this threshold, we can compare their relative effectiveness.

Again, we cannot display much of our data in the present paper, and neither is there a good way of aggregating it, but we can highlight a few interesting results. The top right picture in Fig. 2 is the first in its sequence to reach the 0.0909 threshold, after eight frames. In terms of lens opacity, it looks remarkably like an image employing a different pair of sunglasses, which scores exactly the same. Somewhat less robust than the sunglasses were the nylons shown in Fig. 1, degrading to our threshold after seven frames. A pair of white nylons, which are visibly more transparent than darker nylons, achieved a similar score to the morph frame of the grey nylons with a similar visual opacity. The disguise the proved most robust of all under this analysis used a simple, bright-white party mask, about which we will have more to say in Subsect. 3.4.

### 3.3 The Distance Metric and Performance Trade-Offs

In further research, we would like to see if we can get some leverage directly from the distances that the eigenfaces system returns for each candidate in the image gallery, rather than just analyzing the ordering that this similarity score induces. Unfortunately, a literature search [1,2,6,7,33] indicates that there is no statistical foundation on which we can base a direct interpretation of this "Mahalanobis angle" similarity measure. Indeed, it seems that most of the similarity measures commonly used in the classification literature appeal to some notion of topological distance in the vector space in question, rather than the kind of probabilistic basis that we would wish for in the current study.

Another option is to refine our scoring model to account for the fact we mentioned at the beginning of this section: commercial systems have a way of discarding less likely matches in order to increase the accuracy of the matches, almost always at the expense of an increased rate of false-negatives. Consider Fig. 3: if an operator were to drop results with similarity below 400 or so, he could achieve nearly perfect accuracy. Actually doing so would be foolhardy, however, as our false-negative rate is high enough to make the system essentially useless. More conservative trade-offs are possible, though: the operator can raise accuracy to 76.2% while keeping the false-negatives below 10% using a cut-off of 267. This is still a pretty undesirable false-negative rate, however, so it is not obvious that an operator would make that choice for this system, hence the reason we did not use this scoring system for our present work. We will reconsider this choice when we work with more robust systems, which may have more obvious trade-offs available.

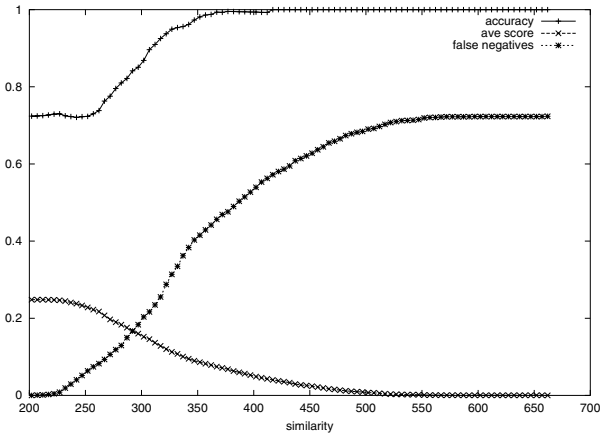



**Fig. 3.** Performance trade-offs

One useful thing this trade-off model would give us, however, is an explanation for the only serious anomaly in our data set: as mentioned before, an image with a scarf wrapped around the lower face scored zero, like many of

the AR images did, however another image with the same scarf wrapped in the same way, with the addition a stocking cap that covers the top of the head (i.e. *more* of the face is actually obscured) actually has a correct match in the first position! Examining the similarity scores of this anomaly, however, reveal that the similarity levels are actually among worst over the face images in the data set, faring even worse than an image of a cat's face, and comparable to a set of probes depicting various inanimate objects, none of which even have correct matches available in the training set. We therefore speculate that this correct answer arose by chance, and this trade-off model would certainly rule it out as such, but we would like to be able to back that claim more rigorously.

### 3.4 Some Modeling Ideas

To conclude this section, we present the beginnings of an explanation that models the success (or lack thereof) of the disguise techniques we developed in the course of this experiment. The countermeasures fell into two broad classes: concealment behind an opaque object, or else overloading of the camera's sensory apparatus with a bright light source. The degree to which each class worked can be summed up in a single word: *contrast*, i.e. large color difference between adjacent pixels. That is, the single most important factor seems to be to add as much contrast as possible, contrast that the system cannot predict based solely on the training photographs, or else to conceal or distract from a contrast that the system expects to find. The reason we think this is the case is that the system is likely to exploit contrast in order to identify candidate discriminating features that it uses to measure (dis)similarity. By adding new areas of contrast, or obscuring existing ones, we can distract the system from the real features, forcing it to map the features it expects to find to the closest coincidental match, ideally a wrong one. The natural world offers many examples that employ camouflage of this nature [9]. Some insects have abdomen coloration that looks like eyes, distracting predators from the real, more vulnerable organs. Also, as is well known, many animals have evolved take advantage of their coloration in order to confuse their features with the background environment.

Recall that some of the AR subjects seem especially hard to disguise. Our preliminary investigation into this fact has yielded some indications that it is helpful to ensure that any reference images of an individual that might end up in a facial recognition database are as feature-deficient as possible. The individuals that had all, or almost all, of their images correctly identified, for instance, all wear glasses in all of their images, and prominent facial hair seems to be a factor as well. As is also the case when one is trying to fool a human observer, it seems likely that successful face recognition countermeasures could include *removing* extra identifying features as well as adding new distractions. Deeper investigation into this phenomenon is ongoing, and we are especially eager to see whether results like this occur in face recognition systems built on foundations other than eigenfaces.

Beyond employing some minimal level of contrast, we think the desired spurious matches are more likely to occur if we insert as many false features as

possible, features not present in the training images. That is, it is better than not to have as many of these contrasts per unit area as we can fit in, provided the contrasting areas themselves are large enough to register as features.

Figure 4 gives some minimal pairs that should illustrate these ideas well. The leftmost image is one of our most robust disguises: a close-fitting party mask. Like our successes with make-up, this mask obscures very little structure of the face, but changes the color profile radically. The next image is nearly the same, only the color of the mask has been digitally modified to be a uniform flesh tone similar to the subject's own coloration. Even though the resulting simulated mask is still opaque, hiding the structure of the face the lies directly behind it, it performs quite poorly, with correct matches in the first, third, and ninth positions. The structure of the mask is the same, but we have all but lost the contrast that its bright color gave us before. The second image from the right is the face paint image that we mentioned before: it also scores a zero (the first correct match is in the sixteenth position). It is interesting to observe that this make-up striping technique resonates well with conventional wisdom on military facial camouflage [26]. Now consider the rightmost image: it employs precisely the same face paint, but covers almost all of the face rather than leaving square gaps that expose the true skin color. Even though we have obscured more of the face, this performs much worse, with a match in the very first position!

**Fig. 4.** Minimal pairs

Our explanation for this is simply that we have eliminated almost all of the contrast that the striping, or the whiteness of the mask, gave to us. Examining the architecture of this eigenfaces system yields an obvious, partial explanation: the system starts by applying a histogram-equalization and scale-normalization filter to the raw images[4], which allows it a degree of robustness over varying

[4] We attempted an experiment that turned off this image processing phase to see if this accounted for the bulk of the performance difference between the striped image and the more fully painted image, however this triggered a bug in the linear algebra library that the system uses, causing it to crash. We have not yet had a chance to investigate fixing this problem.

lighting conditions, and thus likely also global changes in skin tone. Essentially, if a color change is applied globally, the brightness and contrast of the image, as a whole, can be adjusted to give something resembling the same face with the expected color. However, if there are large contrasts in the new color pattern, as is the case with striping, there is no simple way to enhance dark portions of the image without washing out the lighter portions.

A simple way to model our attacks is to divide the face into grid zones, as illustrated in Fig. 5. The size of these zones, at the moment, is somewhat arbitrary, but we think that they should be on order of the size of the major facial features; an experiment is planned to find this critical size threshold. In order to get the best possible results under this model, we simply need to put as large a contrast as we can manage in each zone that doesn't already contain a large contrast, or else we need to hide an existing contrast by removing it entirely (e.g. by shaving off facial hair) or hiding it behind a larger, false contrast. The dark glasses and the white mask, then, hide an expected contrast behind and unexpected, large, distracting contrast, and do so across several zones. The striped face succeeds almost as well by establishing a medium contrast in the majority of facial zones (in terms of color difference, the dark glasses are more than twice as dark as the face paint). Indeed, our model would predict that a darker paint color would perform even better that the dark glasses, and indeed darkening the face paint color digitally achieves exactly the expected result, as did repeating the experiment with more contrastive make-up colors.

**Fig. 5.** Grid model

The model requires a further refinement in order to account for the greater importance of the eye area over the mouth area, as shown in our experiments with the AR images. We can do this by simply weighting the contribution of each score appropriately, which we should be able to do by estimating a probability distribution, based on our current result, for the importance of each zone's contribution, and testing it on novel disguises. Such work is currently in progress.

## 4 Measuring Privacy

This section offers a brief, likely somewhat controversial, foray into modeling privacy problems generally. We believe it to be possible to model all privacy problems, at a high level, in a common framework. In particular, we claim that any privacy problem can be characterized as follows: an adversary knows for certain that a predicate of interest hold of some person in the world, but is uncertain of the identity of that person. For example, the adversary may know that $x_1$ crossed the view of a particular camera lens at time $t_1$, or that $x_2$ bought corn flakes at his store at time $t_2$, or that $x_3$ sent a packet through a mix network that emerged at time $t_3$. The adversary, in attempting to identify the correct valuation of his predicate, builds a probability distribution on the set of all individuals in the world. The job of a privacy enhancing technology is simply to make sure that this probability distribution is not very informative: the correct individual(s) should not stand out.

Our goal in proposing this common framework is to develop a general privacy metric, suitable to serve as the "benefit" half of a cost/benefit ratio, to be used to evaluate any candidate solution meant to mitigate a given privacy problem. A suitable privacy metric should, in principle, be explainable to a mathematically-inclined lay person[5], should be completely orthogonal to any cost metric, and, most importantly, needs to place reliable bounds on how effectively an adversary can unmask an individual trying not to be identified. That is, we want to be able to predict how flat we can make his probability distribution, or place a lower bound on the entropy of his distribution.

The value of a unified model for disparate privacy problems is most clear when one considers multimodal privacy attacks. For example, an adversary may try to correlate multiple biometric sensors, or might try to improve face recognition results using data mining on the purchase data from the store in which the target face image was captured. A common framework will allow us to model such sensor fusion. Also, if the benefit of countermeasures is measured on the same scale, an individual can more easily decide where best to spend his resources when combating more than one privacy problem at once.

Our strategy for identifying a suitable metric is an empirical one: we will propose several candidate metrics and evaluate their effectiveness at predicting the performance of an adversary using several particular attacks. In the work currently in progress, we are concentrating on capturing the limits of biometric HID technologies [20,4], however as our theoretical work continues to evolve, we continually keep in mind other areas of significant privacy concern, such as data mining [11,35] and one of the most widely-studied privacy enhancing technologies, anonymous communication on public networks [25,27,28,29,34].

We conclude this section with a description of one of our most promising candidate metrics. As others have done for anonymity networks [5,31], we draw our inspiration from information theory [32,21]. Consider Fig. 6: we model identity as any rigid identifier, say a unique integer for each individual in the world.

[5] ... at least as explainable, say, as computer performance metrics.

In going about her business, any individual broadcasts this identifier over a noisy channel, a channel which an adversary may be monitoring using one or more sensors. The individual, fortunately, controls some parameters of this noisy channel, while the adversary controls others. The individual wishes to exploit the parameters she controls in order to maximize the entropy in the adversary's probability distribution, which he builds from his sensory information.

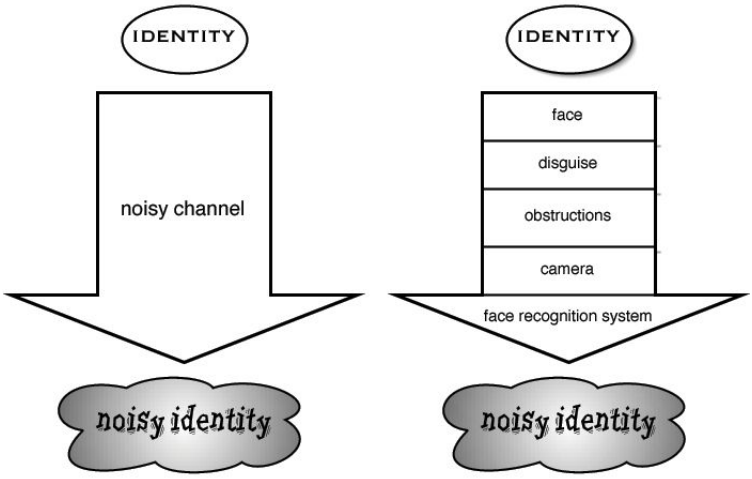



**Fig. 6.** A noisy channel model of privacy problems

Figure 6 also shows a possible instantiation of this model for the face recognition domain. For this problem, the individual's face is a, perhaps imperfect, transmitter for her identity. A camera combined with a face recognition system is her adversary's receiver for this signal. Fortunately, the individual has the options of using obstructions in her environment or actively disguising herself in order to inject more noise into the channel, hoping to limit what the adversary can successfully receive. As is usual in a communication problem, the adversary would like to use his receiver to get the correct information through the channel despite noise. Contrary to the usual case, however, the individual does not wish to cooperate with this communication, and will do her best to disrupt it. Since the individual controls only part of the channel, she cannot direct all of the noise sources to her advantage, but by raising the noise floor, she should be able to directly affect the entropy in the adversary's model.

Figure 7 illustrates how we see our grid model fitting in as part of this noisy channel model. The grid coordinates together with the contents of that grid location make up the symbols in the message being transmitted over the noisy channel, which the adversary would like to decode into an identity. A successful disguise will ensure that not enough signal emerges from the transmitter to be successfully received.

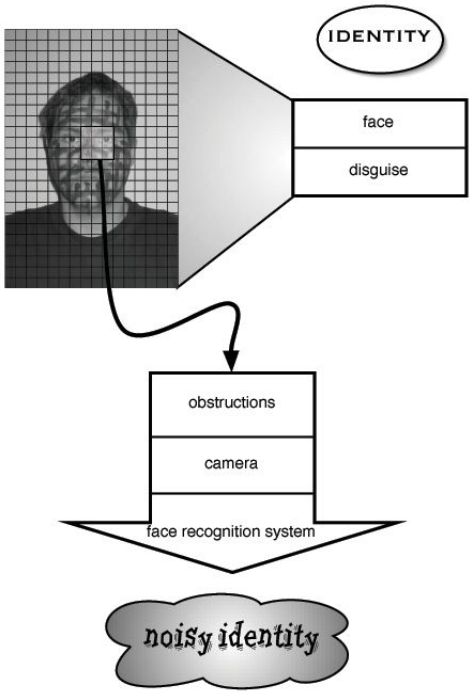



**Fig. 7.** Integration of grid and noisy channel models

A far as we are aware, this noncooperative communicative act is not an aspect of information theory that has been well explored. While it is well-understood how one might continue to successfully communicate over a jammed radio channel, there does not seem to be a theory of how to be an effective jammer, particularly when the jammer is also operating the transmitter. Also, prior models of noise in communication channels assume a randomly-distributed noise source, however our experimental work indicates that the most effective countermeasures require targeted, nonrandom noise. Overcoming these theoretical challenges is a top priority for further research.

## 5 Related Work

Beyond the development of new face recognition algorithms [12,30,36,38,37], the efforts most closely related to our facial recognition research are the FERET test itself [19,23,24] along with the work that followed directly from it [17,3]. A very recent paper [10] made us aware of the existence of the AR database, and itself did a very quick evaluation of the occluded AR images against a different eigenfaces system as well as one of the best commercial systems, FaceIt[6]. Our

[6] We are working on obtaining a license for FaceIt, or a similarly robust system, for evaluation in our own framework.

work differs from these earlier evaluations primarily in perspective: while they are attempting to understand the boundaries of performance of current face recognition system in order to identify performance goals for future research systems, we are trying to find ways of reliably defeating those systems. Consequently, while we and the face recognition community are both interested in understanding why certain faces fail to get identified, our work is additionally interested in those faces that *are* identified when we would rather they were not. This different outlook calls for a significantly different evaluation methodology than pure statistical performance, as well as, for example, trying to find failure modes that those building new systems might decide to ignore, being outside the scope of their intended application.

Although our work is primarily motivated by wanting to prevent or discourage abusive uses of automatic face recognition technology, we certainly recognize that the technology also has perfectly legitimate applications in security and authentication. Indeed, our work should be seen as complementary rather than in opposition to the face recognition community: we expect it to be of value those developing new face recognition techniques and refining existing systems. While tests like those conducted during the FERET trials are useful for understanding how well the technology can perform, in any security application, it is at least as important to understand how and why the technology could fail.

Recent papers from one of the creators of the AR database [15,16] actually focus explicitly on methods of improving performance in the presence of occlusions. Interestingly, his model for working around the occlusions has much in common with our grid model for evaluating occlusions: the face is divided up into zones that are modeled separately, and the output of the individual zone models are combined according to weights generated by a probabilistic likelihood model. We will certainly be interested in evaluating this system, or one like it, using our methodology.

We were recently made aware a preprint [39] of an interesting paper that makes use of the area of information theory we are exploring, channel capacity, in the privacy arena, but with a drastically different scope. It describes an interesting protocol where a consumer can reveal personal information to a market researcher (in exchange for something the consumer wants), but which limits the ability of the marketer to connect accurate information with specific individuals, while maintaining his ability to obtain aggregate information with a known margin of error. Attacks against this protocol are analyzing using Shannon's channel coding theorems.

## 6 Future Directions and Conclusions

The long-term goal of the research program that this paper initiates is to develop a generalized privacy metric. In order to evaluate competing solutions to any problem, engineers must first have a standard of measurement with which to evaluate the candidates. We would like privacy enhancing technology to emerge as a first-class engineering discipline, and a reliable metric is a prerequisite to that.

The present paper establishes some of the empirical foundations on which we intend to build this metric. In particular, it identifies a paradigm of successful countermeasures against one face recognition system, and develops a framework in which we can formalize the crucial properties of those countermeasures without unduly narrowing the scope of our investigation at this early stage.

In the near term, we will continue our investigation into defeating face recognition by expanding our dataset with more disguises applied to more subjects, as well as utilizing reliable synthetic imaging such as the morphing technique described in the present paper. The experiments will, of course, also be replicated using face recognition systems built upon significantly different principles than the one we have already studied. Similar experiments that attempt to counteract other developing HID technologies such as voice, gait [20], and iris recognition [4] will follow, and will be used to further refine and validate our metric.

## Acknowledgments

This research is supported by ONR / DARPA F30602-99-1-0512. Portions of the research in this paper use the FERET database of facial images collected under the FERET program [24]. Thanks to Aleix Martínez for access to the AR database, and Ralph Gross and Jianbo Shi for access to their image annotations as well as valuable discussion.

## References

1. Mark S. Aldenderfer and Roger K. Blashfield. *Cluster Analysis.* Sage Publications, 1984.
2. G. H. Ball. Data analysis in the social sciences: What about the details? In *Proc. AFIPS 1965 Fall Joint Computer Conference*, volume 27, pages 533–559, 1965.
3. J. Ross Beveridge, Kai She, Bruce Draper, and Geof H. Givens. A nonparametric statistical comparison of principal component and linear discriminant subspaces for face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 535–542, 2001.
4. John Daugman. Iris recognition. *American Scientist*, 89(4):326–333, 2001.
5. Claudia Díaz, Stefaan Seys, Joris Claussens, and Bart Prenel. Towards measuring anonymity. In *The Second Workshop on Privacy Enhancing Technologies*, 2002.
6. Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification.* John Wiley and Sons, Inc., 2001.
7. Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, 1972.
8. Simson Garfinkel. *Database Nation.* O'Reilly, 2000.
9. R. L. Gregory and E. H. Gombrich, editors. *Illusion in Nature and Art.* Gerald Duckworth and Co. Ltd., 1973.
10. R. Gross, J. Shi, and J. Cohn. Quo vadis face recognition? In *Third Workshop on Empirical Evaluation Methods in Computer Vision*, 2001.
11. M. A. Hernández and S. J. Stolfo. A generalization of band joins and the merge/purge problem. http://www.cs.columbia.edu/∼sal/hpapers/mpjourn.ps, 1996.

12. Steve Lawrence, C. Lee Giles, Ah Chung Tsoi, and Andrew D. Back. Face recognition: A convolutional neural network approach. *IEEE Transaction on Neural Networks, Special Issue on Neural Networks and Pattern Recognition*, 8(1):98–113, 1997.
13. Alberto Leon-Garcia. *Probability and Random Processes for Electrical Engineering.* Addison-Wesley, 1994.
14. A. M. Martínez and R. Benavente. The AR face database. Technical Report 24, Computer Vision Center, Campus Universitat Autonoma de Barcelona, 1998. http://rvl1.ecn.purdue.edu/ARdatabase/ARdatabase.html.
15. A. M. Martínez. Recognition of partially occluded and/or imprecisely localized faces using a probabilistic approach. In *Proc. of IEEE Computer Vision and Pattern Recognition (CVPR)*, 2000.
16. A. M. Martínez. Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(6):748–763, 2002.
17. Hyeonjoon Moon and P. Jonathon Phillips. Computational and performance aspects of PCA-based face-recognition algorithms. *Perception*, pages 303–321, 2001.
18. Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.
19. NIST. FERET database. http://www.itl.nist.gov/iad/humanid/feret/, 2001.
20. United States General Accounting Office. Technology assessment: Using biometrics for border security. http://www.gao.gov/new.items/d03174.pdf, 2002. Pub. number GAO-03-174.
21. Elements of Information Theory. *Thomas M. Cover and Joy A. Thomas.* Wiley-Interscience, 1991.
22. George Orwell. *1984: a novel.* New American Library, 1961.
23. P. Jonathon Phillips, Patrick J. Rauss, and Sandor Z. Der. FERET (face recognition technology) recognition algorithm development and test results. Technical Report ARL-TR-995, Army Research Laboratory, 1996.
24. P. J. Phillips, H. Wechsler, J. Huang, and P. Rauss. The FERET database and evaluation procedure for face-recognition algorithms. *Image and Vision Computing Journal*, 16(5):295–306, 1998.
25. Charles Rackoff and Daniel R. Simon. Cryptographic defense against traffic analysis. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on the Theory of Computing*, 1993.
26. JV Ramana Rao. *Introduction to Camouflage and Deception.* Defence Research and Development Organization, Ministry of Defense, New Delhi, 1999.
27. Jean-François Raymond. Traffic analysis: Protocols, attacks, design issues and open problems. In Hannes Federath, editor, *Designing Privacy Enhancing Technologies*, Lecture Notes in Computer Science (LNCS 2009), pages 10–29. Springer-Verlag, 2001.
28. Michael Reed, Paul Syverson, and David Goldschlag. Anonymous connections and onion routing. In *IEEE Journal on Selected Areas in Communication Special Issue on Copyright and Privacy Protection*, 1998.
29. Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.
30. F. Samaria and F. Fallside. Automated face identification using hidden markov models. In *Proceedings of the International Conference on Advanced Mechatronics*, 1993.
31. Andrei Serjantov and George Danezis. Towards an information theoretic metric for anonymity. In *The Second Workshop on Privacy Enhancing Technologies*, 2002.

32. Claude E. Shannon and Warren Weaver. *The Mathematical Theory of Communication*. University of Illinois Press, 1949.
33. Roberts R. Sokal and Peter H. Sneath. *Principles of Numerical Taxonomy*. W. H. Freeman and Company, 1963.
34. Paul Syverson, Gene Tsudik, Michael Reed, and Carl Landwehr. Towards an analysis of onion routing security. In *Workshop on Design Issues in Anonymity and Unobservability*, 2000.
35. J. F. Traub and Y. Yemini. The statistical security of a statistical database. *ACM Transactions on Database Systems*, 9(4):672–679, 1984.
36. Matthew A. Turk and Alex P. Pentland. Face recognition using eigenfaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–591, 1991.
37. Dominique Valentin, Hervé Abdi, Alice J. O'Toole, and Garrison W. Cottrell. Connectionist models of face processing: A survey. *Pattern Recognition*, 27:1209–1230, 1994.
38. Dominique Valentin, Hervé Abdi, and Alice J. O'Toole. Categorization of identification of human face images by neural networks: A review of the linear autoassociative and principal component approaches. *Journal of Biological Systems*, 2(3):413–429, 1994.
39. Poorvi L. Vora. Towards a theory of variable privacy. In review, available at http://www.hpl.hp.com/personal/Poorvi_Vora/Pubs/plv_variable_privacy.pdf, 2002.
40. Matthew Wright, Micah Adler, Brian N. Levine, and Clay Shields. An analysis of the degradation of anonymous protocols. In *Proceedings of the ISOC Network and Distributed System Security Symposium (NDSS 2002)*, 2002.

# From Privacy Legislation to Interface Design: Implementing Information Privacy in Human-Computer Interactions

Andrew S. Patrick[1] and Steve Kenny[2]

[1] Institute for Information Technology, National Research Council of Canada
Building M-50, 1200 Montreal Rd., Ottawa, ON Canada K1A 0R6
Andrew.Patrick@nrc-cnrc.gc.ca
[2] Independent Consultant
stephen_mh_kenny@yahoo.com

**Abstract.** Internet users are becoming more concerned about their privacy. In addition, various governments (most notably in Europe) are adopting strong privacy protection legislation. The result is that system developers and service operators must determine how to comply with legal requirements and satisfy users. The human factors requirements for effective privacy interface design can be grouped into four categories: (1) comprehension, (2) consciousness, (3) control, and (4) consent. A technique called "Privacy Interface Analysis" is introduced to show how interface design solutions can be used when developing a privacy-enhanced application or service. To illustrate the technique, an application adopted by the Privacy Incorporated Software Agents consortium (PISA) is analyzed in which users will launch autonomous software agents on the Internet to search for jobs.

## 1 Introduction

### 1.1 Project Motivations and Goals

There is increased awareness by the general public of their right to, and the value of, their privacy. Recent surveys indicate that Internet users are very concerned about divulging personal information online, and worried that they are being tracked as they use the Internet [8]. Research has indicated that users are failing to register for WWW sites because they feel that they cannot trust the Internet with personal or financial information [14]. In addition, information privacy is increasingly being associated with business issues such as reputation and brand value [6]. Moreover, governments within the European Union, Canada, Australia, and Switzerland have adopted privacy protection legislation that is enforced through independent governmental bodies with significant oversight powers. There has been little guidance, however, provided to system developers and operators on how to implement and comply with these privacy guidelines and rules, and how to soothe users' privacy concerns. This paper is an attempt to fill that gap.

This work was conducted as part of the Privacy Incorporated Software Agents (PISA; www.pet-pisa.nl) project, a European Fifth Framework Programme project whose goal is to develop and demonstrate Privacy-Enhancing Technologies (PET) that will protect the privacy of individuals when they use services that are imple-

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 107–124, 2003.

mented through intelligent software agents. An integral part of the project is an analysis of the European privacy legislation and the development of methods to translate legislative clauses into human-computer interaction (HCI) implications and interface specifications. HCI is the study of mental processes and behavior as they pertain to users interacting with computers (and other technical devices). The goal of this paper is to document a process that begins with privacy legislation, works through derived privacy principles, examines the HCI requirements, and ends with specific interface design solutions. The approach taken is one of "engineering psychology" in which knowledge of the processes of the brain is used when doing system design [18].

In the sections that follow we explain how the European Privacy Directive 95/46/EC [3] has been analyzed to produce a set of detailed privacy principles (Section 2). The principles are then examined from a human factors point of view and a set of HCI requirements are developed (Section 3). We then demonstrate how the HCI requirements can be used when planning or analyzing a software application or service (a process we call a "Privacy Interface Analysis"; Section 4). Overall, our intent is to introduce the core concepts of privacy protection and HCI requirements, and then illustrate a Privacy Interface Analysis that other developers can follow.

To illustrate the technique, we use an example application adopted by the PISA consortium. This example is a computer service in which users will launch autonomous software agents on the Internet to search for jobs. The agents will have personal information about the users that the agents will use when seeking appropriate placements with various employers, so protection of the users' privacy is important. In the PISA demonstrator, each user has a personal agent to which he can delegate tasks such as searching for a job or making an appointment with another person or company. The personal agent in turn creates a dedicated agent for each task it is given. For example, a Job Search Agent (JSA) might communicate with Market Advisor Agents to locate good places to look for jobs. A Job Search Agent may also interact with a Company Agent to get more information about a position. Maintaining privacy protection as the agents share information and make autonomous decisions is the challenge of the PISA project.

### 1.2 Related Work

Alfred Kobsa [7][8] has recently conducted analyses with goals similar to the current project. Kobsa is interested in personalization services, such as WWW sites that remember your name and preferences. Such personalized services are made possible because the sites collect personal information about the users, either explicitly by asking for the information, or implicitly by tracking usage patterns. Although the personalized services can be useful and valuable, the storage and use of personal information both worries some users, and falls under the auspices of privacy guidelines and legislation. Kobsa has examined the implications of the privacy laws and user concerns and developed design guidelines to help WWW site operators build privacy-sensitive systems. These guidelines include suggestions like: (1) inform users that personalization is taking place, and describe the data that is being stored and the purpose of the storage, (2) get users' consent to the personalization, and (3) protect users' data with strong security measures. The current analysis goes deeper to focus on the requirements necessary for complying with the European Privacy Directive, and includes a discussion of specific interface techniques that can be used to meet those requirements.

# 2 Privacy Principles

## 2.1 EU Legislation

The right to privacy in the EU is defined as a human right under Article 8 of the 1950 European Convention of European Human Rights. The key privacy document is Directive 95/46/EC of the European Parliament on the protection of individuals with regard to the processing of personal data, and the free movement of such data (hereafter referred to as The Directive) [3]. Also, Directive 97/66/EC [4], concerning the processing of personal data and the protection of privacy in the telecommunications sector, applies and strengthens the original directive in the context of data traffic flow over public networks. These two directives represent the implementation of the human right to privacy within the EU.

The Directive places an obligation on member states to ratify national laws that implement the requirements of The Directive. This has resulted in, for instance, Wet Bescherming Persoonsgegevens 1999 in The Netherlands and The Data Protection Act 1998 in the UK. The national legislatures of EU member states must implement The Directive to substantially similar degrees. Such implementation includes sanctioning national enforcement bodies such as the Dutch Data Protection Authority with prosecution powers.

The Directive defines a set of rights accruing to individuals concerning personal data (also known as Personally Identifiable Information, or PII), with some special exceptions, and lays out rules for lawful processing of that information that are applicable irrespective of the sector of application. Specifically, The Directive specifies the data protection rights afforded to citizens or "data subjects", plus the requirements and responsibilities of “data controllers” and by association “data processors”. The Directive attempts to balance the fundamental right to privacy against the legitimate interests of data controllers and processors -- a distinctive and central characteristic of the EU approach to data protection.

## 2.2 Overview of the Resulting Principles

As The Directive concerns itself with data processing, it must be implemented through a combination of information technology and governance initiatives. Privacy principles abstracted from the complexities of legal code have been developed to simplify this process. Table 1 shows a high-level summary of the privacy principles. Our research has focused on the privacy principles of (1) transparency, (2) finality and purpose limitation, (3) lawful basis, and (4) rights because these principles have the most important implications for user interface design. The remainder of this paper will be restricted to these four privacy principles.

# 3 HCI Requirements

## 3.1 Deriving the Requirements

The principles shown in Table 1 have HCI implications because they describe mental processes and behaviors that the Data Subject must experience in order for a service to adhere to the principles. For example, the principles require that users *understand*

the transparency options, are *aware* of when they can be used, and are able to *control* how their PII is handled. These requirements are related to mental processes and human behavior, and HCI techniques are available to satisfy these requirements. For example, an HCI specialist might examine methods for ensuring that users understand a concept, such as providing documentation, tutorials, and interface design characteristics.

**Table 1.** High-Level Summary of Privacy Principles (italic items are analyzed in detail).

| Principle | Description |
|---|---|
| Reporting the processing | All non-exempt processing must be reported in advance to the National Data Protection Authority. |
| *Transparent processing* | *The Data Subject must be able to see who is processing his personal data and for what purpose. The Controller must keep track of all processing performed by it and the data Processors and make it available to the user.* |
| *Finality & Purpose Limitation* | *Personal data may only be collected for specific, explicit, legitimate purposes and not further processed in a way that is incompatible with those purposes.* |
| *Lawful basis for data processing* | *Personal data processing must be based on what is legally specified for the type of data involved, which varies depending on the type of personal data.* |
| Data quality | Personal data must be as correct and as accurate as possible. The Controller must allow the citizen to examine and modify all data attributable to that person. |
| *Rights* | *The Data Subject has the right to acknowledge and to improve their data as well as the right to raise certain objections.* |
| Data traffic outside EU | Exchange of personal data to a country outside the EU is permitted only if that country offers adequate protection. If personal data is distributed outside the EU then the Controller ensures appropriate measures in that locality. |
| Processor processing | If data processing is outsourced from Controller to Processor, controllability must be arranged. |
| Security | Protection must be provided against loss and unlawful processing. |

Table 2 (in the Appendix) presents a more detailed summary of the four privacy principles under consideration in this paper. Included in Table 2 are the HCI requirements that have been derived from the principles. These requirements specify the mental processes and behavior of the end user that must be supported in order to adhere to the principle. For example, the principle related to the processing of transparency leads to a requirement that users know who is processing their data, and for what purpose.

The HCI requirements outlined in Table 2 are not unrelated. The core concepts in the requirements can be grouped into four categories: (1) *comprehension*: to understand, or know; (2) *consciousness*: be aware, or informed; (3) *control*: to manipulate, or be empowered; (4) *consent*: to agree.

In the category of *comprehension*, the requirements can be summarized as building a system or service that will enable users to:

- comprehend how PII is handled
- know who is processing PII and for what purposes

- understand the limits of processing transparency
- understand the limitations on objecting to processing
- be truly informed when giving consent to processing
- comprehend when a contract is being formed and its implications
- understand data protection rights and limitations

In the category of *consciousness*, the requirements are to allow users to:

- be aware of transparency options
- be informed when PII is processed
- be aware of what happens to PII when retention periods expire
- be conscious of rights to examine and modify PII
- be aware when information may be collected automatically

In the category of *control*, the requirements are to allow users to:

- control how PII is handled
- be able to object to processing
- control how long PII is stored
- be able to exercise the rights to examine and correct PII

Finally, the requirements in the area of *consent* are to build systems that allow users to:

- give informed agreement to the processing of PII
- give explicit permission for a Controller to perform the services being contracted for
- give specific, unambiguous consent to the processing of sensitive data
- give special consent when information will not be editable
- agree to the automatic collection and processing of information

This list represents the essential HCI requirements that must be met in order to build systems that provide usable compliance with the European Privacy Directive. System designers will be well served if they consider the dimensions of comprehension, consciousness, control and consent when building privacy-enhanced systems.

### 3.2 Interface Methods to Meet Requirements

The field of interface design has developed a set of techniques, concepts, and heuristics that address each of the requirement areas. It is beyond the scope of this paper to provide an exhaustive review of the field of interface design, and interested readers are encouraged to examine one of the many HCI books for more information [e.g., 15, 11, 10, 18, 12].

**Comprehension.** The obvious method to support comprehension or understanding is training. Users can be taught concepts and ideas through classroom training, manuals, demonstrations, etc. Such methods can be very successful, but they can also be expensive, time-consuming, and inappropriate when learning computer systems that will be used infrequently. Today, much effort is devoted to supporting comprehension without resorting to formal training methods.

User documentation, especially online or embedded documentation, is often used as a replacement for training. Most computers and software come with manuals of some sort, and much is known about how to develop material that people can learn from effectively [10]. Studies have shown, however, that most users do not read the

documentation, and often they cannot even find the printed manuals [1]. As a result, designers often resort to tutorials and help systems to support comprehension. Help systems can be designed to provide short, targeted information depending on the context, and such systems can be very powerful. It is often difficult, however, to learn an overview of all the features of a system using built-in help. Tutorials are another method of supporting learning, and they can work well if they are designed with a good understanding of the needs of the user.

There are other methods for supporting understanding that do not rely on documentation. For example, research in cognitive psychology has shown that users often develop personal "mental models" of complex systems. These models are attempts to understand something to a level where it can be used effectively, and such models can be quite effective when faced with complex systems. HCI specialists can exploit the human tendency to create models by either guiding users to develop appropriate models, or by examining the models that already exist and accounting for them. For example, people often have a mental model of a furnace thermostat that is analogous to a water faucet. That is, the more that it is "turned on", the faster the water (or heat) will flow. This model is incorrect because most furnaces can only operate at one flow rate and the thermostat only determines the temperature where the heat flow will be shut off. It is interesting to note that this erroneous mental model has persisted for a long time, and thermostat interface designers would likely want to take it into account. Thus, a thermostat designer might add a feature to automatically return the setting to a normal room temperature some time after the thermostat was suddenly turned to an abnormally high setting.

A related interface technique is the use of metaphors. Most modern graphical computer systems are based on a desktop or office metaphor, where documents can be moved around a surface, filed in folders, or thrown in a trashcan. The graphical elements of the interface, such as document icons that look like pieces of paper and subdirectory icons that look like file folders, reinforce this metaphor. The metaphor is valuable because it provides an environment that users are familiar with, and thus they can use familiar concepts and operations when interacting with the system. The familiar metaphor decreases the need to develop new knowledge and understanding.

There are other, more subtle techniques that can facilitate comprehension. For example, the layout of items on the screen can convey some meaning or information. Items that are grouped together visually will likely be considered to be group together conceptually [10], and interface designers can take advantage of that. Also, items that are ordered horizontally in a display will likely be examined from left to right, at least in North American and European cultures. Interface designers can use this sequencing tendency to ensure that users follow the recommended order of operations.

Feedback is also very important for supporting understanding [15]. Most complex systems require some experience and learning before they can be used effectively. Without feedback, users may not learn the consequences of their actions and understanding will be slow to develop.

**Consciousness.** The requirement of consciousness refers to the user being aware of, or paying attention to, some concept or feature at the desired time. It is related to comprehension because the awareness may require some background knowledge before conscious attention is useful. Consciousness in this context can be thought of as bringing knowledge or understanding to the attention of the user so it can be used when required.

There are many interface techniques for making users aware of something. System messages or pop-up windows are an obvious technique for making the user aware of an event. For important information, these windows can be constructed so the users have to acknowledge the message before they can continue using the system. A more subtle technique is to remind the user of something without interrupting their work. This is sometimes seen in "help assistants" (such as the Microsoft Office Assistant) that make suggestions while users interact with the interface. Another way to remind users is through the arrangement of the interface. For example, if a particular option is available to a user at a certain time, placing icons or messages nearby in the interface layout can ensure that users are aware of the options.

Even more subtle methods use display characteristics to draw attention. Printing text in a certain color, such as red, can draw attention. Changing the color dynamically can be more effective. Sounds are also frequently used to make users aware of some event. The human factors discipline has a long history of designing systems that make users aware of certain things at certain times [18].

**Control.** Control refers to the ability of the user to perform some behavior. Control is related to comprehension because the user must understand the task and context to behave effectively. Control is also related to consciousness because users must be aware of the need to act before they can execute the behavior. The issue of control, however, is that once the user knows that they are supposed to do something (awareness), and they understand what to do (comprehension), can they actually carry out the action.

**Fig. 1.** A door with poor affordances. The door is solid glass with a vertical handle in the middle. (from http://www.baddesigns.com; reprinted with permission).

An important concept for ensuring control is affordance, which means to provide naturally or inevitably. The classic example is door opener design. With some doors, users may approach the door, understand that it is a door, be conscious that they need to open the door, and still not be able to perform the action (see Figure 1 for an example). In contrast, a simple metal plate placed on the surface of the door tends to be a natural signal to push the door (in fact, these are often called "push plates"), whereas a metal loop placed vertically at the edge of a door tends to be a natural signal to pull the door. By using affordances, interface designers can make the door easy to control.

Another interface technique that supports appropriate actions is mapping. The idea is to map the appearance and function of the interface to the device being controlled. This might mean making a physical analogy of the real world in the interface, such as arranging light switches on a wall in the same order that the lights are arranged in the ceiling [11].

Many of the subtle HCI techniques that can be used to support control are related to "obviousness". To the extent that the interface can be made obvious to the user,

control (and understanding) can be smooth and effective. When interfaces are not obvious, users may have serious problems using the device or system. The goal of the interface designer is to build something that is so obvious to the user that comprehension, consciousness, and control will develop with little learning and effort.

**Consent.** The final HCI requirement category is consent. Users must be able to consent or agree to terms or conditions that may be associated with a system or service. Moreover, the consent should be "informed", meaning that the users fully understand what they are agreeing to, and what implications this may have. Obviously, supporting informed consent is related to the requirements for comprehension and consciousness.

The most common method for supporting consent in computer applications is a "user agreement". When you have installed new software on your computer, or signed-up for an Internet service, you have undoubtedly seen an interface screen that presents a User Agreement or Terms of Service. In order to continue, you have had to click on an "I Agree" button or an equivalent label. These interface screens are commonly called "click-through agreements" because the users must click through the screen to get to the software or service being offered [17]. (An alternative label is "click-wrap agreement", in parallel to more traditional "shrink-wrap" agreements attached to software packaging.) These agreement screens are an attempt to provide the electronic equivalent of a signed user agreement or service contract [16]. By clicking on the "Agree" button, the user is confirming their understanding of the agreement and indicating consent to any terms or conditions specified in the accompanying text.

The legality of these click-through screens in forming the basis of a legal agreement or contract has been established, but with some qualifications. The Cyberspace Law Committee of the American Bar Association has recently reviewed the case law and developed a set of guidelines for creating click-through agreements [9]. These guidelines have been summarized into six principles to be considered by system developers [5][17]:

1. Opportunity to review terms: users must view the terms of the agreement before consenting to the agreement. A recent case involving Netscape [17] established that it is important that there be no other method to obtain the product or service other than by clicking-through the agreement.
2. Display of terms: the terms have to be displayed in a "reasonably conspicuous" [17] manner. A recent case involving Ticketmaster [9] established that simply linking to the terms at the end of a long home page was not enough.
3. Assent to terms: the language used to accept the agreement must clearly indicate that a contract is being formed.
4. Opportunity to correct errors: there should be a method for users to correct errors, such as seeking a final confirmation before proceeding, or allowing the user to back-out of an agreement.
5. Ability to reject terms: the option to reject the terms of the agreement should be clear and unambiguous, and the consequences of the rejection should be stated (e.g., "if you do not agree, you will not be able to install this software").
6. Ability to print the terms: the interface should allow the user to print the terms for later reading.

Other factors that should be considered when creating click-through agreements [16] are to redisplay the terms and conditions at product startup (reminding), and to support the ability to review the terms at any time (e.g., in the "help" or "about" menus). In addition, developers should adapt the terms and conditions to local languages and requirements. If these principles and considerations are heeded, case law suggests that click-through agreements will likely be enforced, at least in US courts. (Some jurisdictions, such as Germany and China, are unlikely to enforce any of these agreements [16]).

The text of many click-through agreements tends to be long and complex, often to ensure that all the points raised above are addressed. The result is that many users have difficulty reading and understanding the documents (a comprehension problem), and many users click the "Agree" button without considering the terms at all (a consciousness problem). The problems arise because people have limited cognitive capacity: we have limited attention spans, a restricted ability to process large quantities of detailed information at one time, and limited memories. Thus, using interface techniques that are sensitive to user characteristics may be valuable here. This observation may be particularly relevant if users are being asked to agree to a number of terms that will affect them substantially, such as the processing of their personal data.

Ensuring that users fully understand and unambiguously agree to the processing of their personal information is important for complying with privacy legislation and guidelines. Consider the definition of consent provided in the EU Directive 95/46/EC on privacy protection [3]:

> 'the data subject's [user's] consent' shall mean any freely given specific and informed indication of his wishes by which the data subject signifies his agreement to personal data relating to him being processed. (Article 2-h)

It is clear that a large, cumbersome, complicated User Agreement presented to the user only when they begin to use a product or service fails to live-up to the requirements for "specific" and "informed" consent, and yet these types of user agreements are the majority. These issues are of particular concern in relation to explicit consent. For example, the EU Directive states that when sensitive data (e.g., race, ethnic origin, religious beliefs) are processed, the user must give "explicit consent" (Article 8-2-a) to the processing of the sensitive data. Again, a single, large, click-through User Agreement does not meet the spirit of The Directive.

The solution to this problem proposed here is a new concept of "Just-In-Time Click-Through Agreements" (JITCTAs). The main feature of a JITCTA is not to provide a large, complete list of service terms but instead to confirm the understanding or consent on an as-needed basis. These small agreements are easier for the user to read and process, and facilitate a better understanding of the decision being made in-context. Also, the JITCTAs can be customized for the user depending on the features that they actually use, and the user will be able to specify what terms they agree with, and those they do not. It is hoped that users will actually read these small agreements, instead of ignoring the large agreements that they receive today. The responses made by the user during the JITCTAs can also be recorded so there is a clear, unambiguous record of the specific agreements made with the user. In order to implement JITCTAs, the software will have to recognize when users are about to use a service or feature that requires that they understand and agree to some term or condition.

A sample screen capture of a JITCTA is shown in Figure 2. In this example a user has selected the Trade Union Membership information field in the Create Agent interface screen of the PISA interface. Since this would be considered sensitive information in the EU Privacy Directive, a JITCTA has appeared to obtain explicit, specific, timely, unambiguous consent to the processing of this data.

In summary, well-formulated click-through agreements are legally permissible in many countries, and Just-In-Time Click Through Agreements improve on this device by supporting more appropriate decision-making and control that is sensitive to human factors constraints.

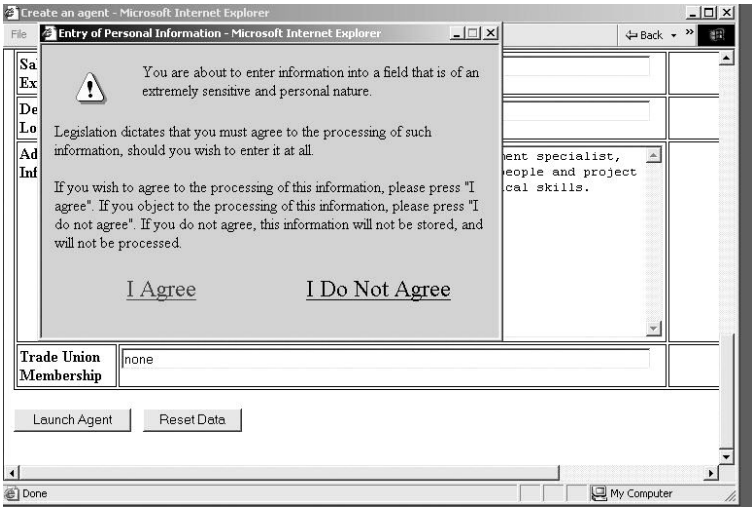



**Fig. 2.** An example of a Just-In-Time Click-Through Agreement (JITCTA).

## 4 The Privacy Interface Analysis

Up until this point, we have described the privacy principles that have been derived from the European Privacy Directive, analyzed these principles for their HCI requirements, categorized and described the nature of the requirements, and reviewed methods to meet these requirements. This section outlines how all of this can be brought together to systematically conduct a Privacy Interface Analysis.

### 4.1 Develop a Service/Application Description

The first step in the analysis is to prepare a detailed description of the operation of the program or service. A useful technique for conducting this analysis is the Unified Modeling Language (UML) [13], which is a powerful language for specifying, visualizing, and sharing specifications and design decisions. By creating a set of interrelated diagrams or models, the developers can visualize and examine the features of the software long before any programming code is written. Although UML is not required to complete a thorough privacy interface analysis, it does make the process easier and the result more valuable.

A primary UML modeling technique is Use Case modeling. Here a high-level diagram is created to show the functionality of the system from the users' point of view. The purpose of the Use Case analysis is to specify what the software will do, and not to focus on how it will do it (that will come later). Figure 3 shows a simple Use Case diagram for the PISA Demonstrator example. This diagram shows the major functions provided by the software are creating an agent, tracking an agent, viewing agent results, etc. Doing a thorough analysis at this stage is important because each use case represents a function or feature that may involve an interface to privacy protection measures.

The next step is to determine how the application will work internally. UML structure diagrams are useful here to illustrate the software objects or classes that will be necessary to implement the functionality of a use case. Perhaps most useful are interaction diagrams, such as Object Sequence Diagrams. These diagrams model the relations between the software objects, and illustrate any data communication that must take place. Figure 4 shows a sequence diagram for the Register use case in the PISA demonstrator example. This diagram depicts the major software components involved with supporting this function, such as the WWW interface, the WWW server, and the Personal Agent. It also shows the interactions between the user and the system, as well as the interactions between the software objects. Normally you should create at least one Object Sequence diagram for each use case that was identified earlier.

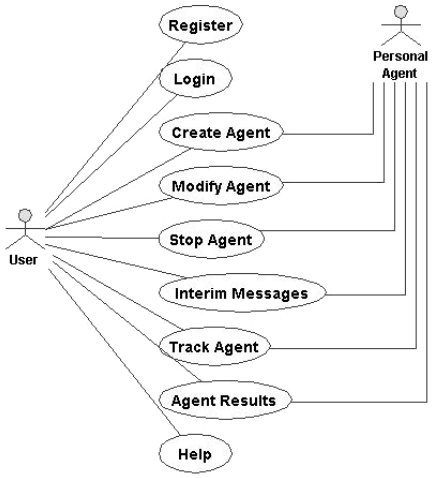



**Fig. 3.** Use Case Diagram for the PISA Demonstrator.

### 4.2 Explore and Resolve the HCI Requirements

The third step involves analyzing the HCI requirements of each of the privacy principles in Table 2 and determining their effects on the application models. For each principle, determine if the human requirements related to the principle are already covered in the current models of the application, or if a solution is required. If a solution is needed, generic possible solutions to the HCI requirements are presented in the last column of Table 2, but each application may require a unique solution that is suitable for that particular situation. For example, Principle 1.3.1 concerns processing for direct marketing purposes, and states that: "DS receives notification of possible objection". Applied to the PISA example, this means that users need to be made aware that they are able to object to processing of personal data for direct marketing purposes (the comprehension and consciousness requirement categories). One method to satisfy this principle would be to include an "opt-in" feature in the Create

Agent use case so users can choose to participate in direct marketing or not, and to display that option in a distinctive color to draw attention to it. In addition, a "review options" function might be added to the Modify Agent use case to remind users that they can view and change their opt-in decision. Also, in the Track Agent use case, a control to change their opt-in decision could be provided.

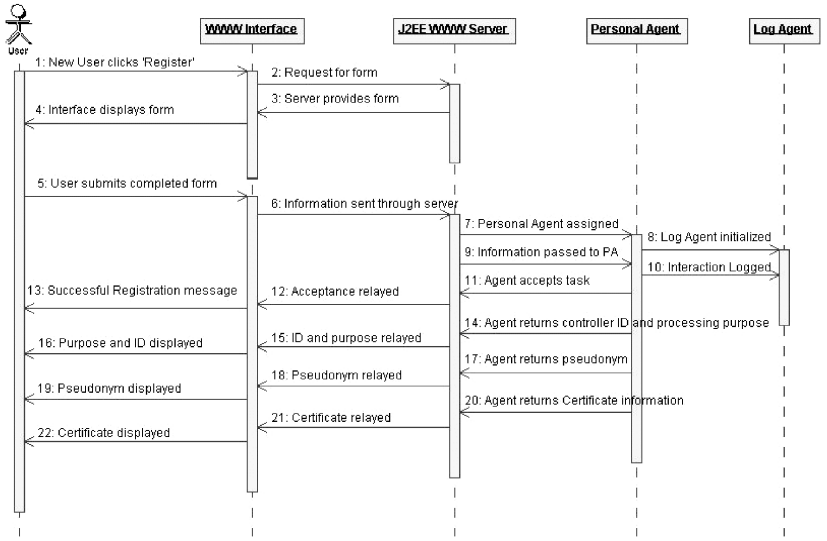



**Fig. 4.** Object Sequence Diagram for the Register Use Case.

To further illustrate this step in the analysis, consider what must happen during the Create Agent use case. A naive view might be that the user simply provides the system with personal information, and perhaps reads a user agreement. By applying the HCI requirements, this Create Agent function can be expanded to ensure usable compliance with the privacy principles. For example, Principle 2.3 states that personal information must have an associated retention period, after which the data is deleted or rendered anonymous. To comply with this requirement, an interface feature to "specify retention period" can be added to the Create Agent use case. Other features that should be included in the Create Agent use case are:

- use a JITCTA to acknowledge rights
- use a JITCTA to acknowledge the formation of a contract and to consent to PII processing
- use a JITCTA if any sensitive information is collected
- provide an interface to "opt-in" to processing for direct marketing purposes

Another example of the results of a privacy interface analysis is shown in Figure 4. Principle 1 states that the use and storage of PII must be transparent to the user. To meet that requirement, the interaction diagrams were examined and extra interactions for the Register use case were added so information about the identity and purpose of the Controller are conveyed to the user.

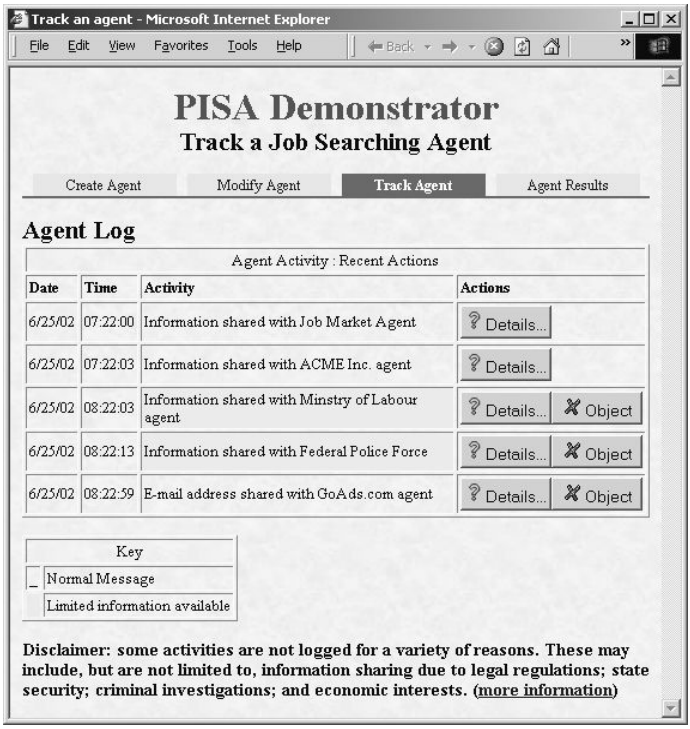



**Fig. 5.** A possible Track Agent interface screen illustrating HCI solutions.

Another important HCI requirement is that users must understand their ability to track the processing of their PII, and be aware of any limitations. In the PISA example, a solution to this requirement is shown in Figure 5, which represents a possible Track Agent interface screen. This screen shows how a log of agent information sharing could be displayed, and some log entries are highlighted to indicate that limited tracking information is available. In addition, users are reminded by the message at the bottom of the screen of the situations where activity may not have been logged at all. Another feature of the interface is to place control buttons for the objection functionality alongside the appropriate log entries. Thus, by using the interface features of highlighting, reminding, and grouping, the privacy principles can be implemented naturally and obviously.

The result of a well-conducted privacy interface analysis is a set of design solutions that will ensure usable compliance with the privacy principles. These can be organized according to the use cases that are affected and incorporated into a product design specification and passed on to the developers for implementation.

### 4.3 Conducting a Privacy Interface Analysis for Other Applications

Developers interested in conducting a Privacy Interface Analysis should now be ready to proceed. Again, the key steps are to:

1. develop a detailed description of the application or service from a use case and internal operation point of view.
2. examine each HCI requirement described in Section 3.1 to see if it applies to this application, using Table 2 as a guide.
3. for each requirement that must be met, scrutinize the generic privacy solutions provided in Table 2 (and the interface design methods in Section 3.2) to determine an appropriate specific solution.
4. organizing the solutions according to use cases and capture the solutions in an interface requirements document.
5. implement the interface according to the requirements document.

## 5 Summary and Conclusions

This paper introduced design guidance for privacy-enhancing technologies from a human factors point of view. For the first time, this work specified what must be included in human-computer interfaces to satisfy the spirit of European privacy legislation and principles, and satisfy the privacy needs of the users ("usable compliance"). A technique called "privacy interface analysis" was introduced to help developers establish the privacy requirements for their projects, and understand the interface design solutions that can be used.

The current work has focused on European privacy legislation and, although the resulting principles, requirements, and solutions are general, one of the challenges that remains is to ensure that the knowledge is equally applicable in other legislative settings, such as Canada, and in areas operating in a self-regulatory fashion (e.g., the USA). For example, it is possible that the market forces operating in the USA will lead to privacy requirements and expectations that have not been anticipated. Even in regulated environments, the privacy legislation and guidelines will change and evolve, and thus the human interface guidelines will also have to be dynamic.

Privacy enhancing technologies are also evolving and changing, and this will have an effect on the types of solutions that are available, and also the privacy needs and expectations of the users. For example, the P3P protocol, if implemented widely, may have a profound effect on the privacy domain by bringing privacy issues to the attention of millions of Internet users, and hopefully providing an easy-to-use privacy control interface (e.g., [2]).

Our research is continuing in this area. We will use the techniques introduced here during the completion and evaluation of the PISA prototype. Usability studies being conducted now will provide concrete data on the effectiveness of interface design solutions proposed here in meeting users' privacy needs. We are also beginning to examine the process of developing and implementing privacy policies, where we are also interested in the steps required when moving from intentions, to principles, to requirements, and to implementations.

## References


1. Comstock, E.M., & Clemens, E.A. (1987). Perceptions of computer manuals: A view from the field. *Proceedings of the Human Factors Society 31st Annual Meeting*, 139-143.
2. Cranor, L.F., Arjula, M., & Guduru, P. (2002). Use of a P3P User Agent by Early Adopters. *Proceedings of Workshop on Privacy in the Electronic Society*. Washington, D.C., November 21.
3. Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the Protection of Individuals with Regard to the Processing of Personal Data and on the Free Movement of such Data. *Official Journal of the European Communities* (1995), p. 31.
4. Directive 97/66/EC of the European Parliament and of the Council of 15 December 1997 concerning the processing of personal data and the protection of privacy in the telecommunications sector. *Official Journal L 024* , 30/01/1998 p. 0001 – 0008.
5. Halket, T.D., & Cosgrove, D.B. Is your online agreement in jeopardy? http://www.cio.com/legal/edit/010402_agree.html
6. Kenny, S., & Borking, J. (2002). The value of privacy engineering. *Journal of Information, Law and Technology (JILT)*. http://elj.warwick.ac.uk/jilt/02-1/kenny.html.
7. Kobsa, A. (2001). Tailoring privacy to users' needs (Invited Keynote). In M. Bauer, P. J. Gmytrasiewicz and J. Vassileva, Eds. *User Modeling 2001: 8th International Conference*. Berlin - Heidelberg: Springer Verlag, 303-313. http://www.ics.uci.edu/~kobsa/papers/2001-UM01-kobsa.pdf
8. Kobsa, A. (2002). Personalized hypermedia and international privacy. *Communications of the ACM, 45*(5), 64-67. http://www.ics.uci.edu/~kobsa/papers/2002-CACM-kobsa.pdf
9. Kunz, C.L. (2002). Click-Through Agreements: Strategies for Avoiding Disputes on Validity of Assent. http://www.efscouncil.org/frames/Forum%20Members/Kunz_Click-thr_%20Agrmt_%20Strategies.ppt. See also C.L. Kunz, J. Debrow, M. Del Duca, and H. Thayer, "Click-Through Agreements: Strategies for Avoiding Disputes on Validity of Assent," *Business Lawyer, 57,* 401 (2001).
10. Nielsen, J. (1993). *Usability engineering*. San Diego, CA: Morgan Kaufmann.
11. Norman, D.A. (1988). *The psychology of everyday things*. Basic Books.
12. Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., & Carey, T. (1994). *Human-computer interaction*. Reading, MA: Addison-Wesley.
13. Rumbaugh, J., Jacobson, I., & Booch, G. (1998). *The unified modeling language reference manual*. Addison-Wesley.
14. Saunders, C. (2001). Trust central to E-commerce, online marketing. *Internet Advertising Report*. http://www.internetnews.com/IAR/article.php/12_926191
15. Shneiderman, B. (1987). *Designing the user interface: Strategies for effective human-computer interaction*. Reading, MA: Addison-Wesley.
16. Slade, K.H. (1999). Dealing with customers: Protecting their privacy and enforcing your contracts. http://www.haledorr.com/db30/cgi-bin/pubs/1999_06_CLE_Program.pdf
17. Thornburgh, D. (2001). Click-through contracts: How to make them stick. *Internet Management Strategies*. http://www.loeb.com/FSL5CS/articles/articles45.asp
18. Wickens, C.D., & Hollands, J.G. (2000). *Engineering psychology and human performance* (3rd Ed.). Upper Saddle River, NJ: Prentice Hall.

# Appendix

**Table 2.** Privacy Principles, HCI Requirements, and Design Solutions.

| | **Privacy Principle** | **HCI Requirement** | **Possible Solution** |
|---|---|---|---|
| 1 | Transparency: Transparency is where a Data Subject (DS) is empowered to comprehend the nature of processing applied to her personal data. | users must be *aware* of the transparency options, and feel empowered to *comprehend* and *control* how their Personally Identifiable Information (PII) is handled | during registration, transparency information is *explained* and examples or tutorials are provided |
| 1.1 | DS informed: DS is aware of transparency opportunities | users must be *aware* of the transparency options | Opportunity to track controller's actions made *clearly visible* in the interface design |
| 1.1.1 | For: PII collected from DS. Prior to PII capture: DS informed of: controller Identity (ID) and Purpose Specification (PS) | users *know* who is controlling their data, and for what purpose(s) | at registration, user is *informed* of identity of controller, processing purpose, etc. |
| 1.1.2 | For: PII not collected from DS but from controller. DS informed by controller of: processor ID and PS. If DS is not informed of processing, one of the following must be true: DS received prior processing notification, PS is legal regulation, PS is security of the state, PS is prevention/detection/prosecution of criminal offences, PS is economic interests of the state, PS is protection of DS or rights of other natural persons, PS is scientific/statistical & PII is anonymized, or PII are subject to any other law governing their processing/storage | users are *informed* of each processor who processes their data, and the users *understand* the limits to this informing | - *user agreements* states that PII can be passed on to third parties<br>- user agreement also contains information about usage tracking limitations<br>- when viewing the processing logs, entries with limited information are coded to draw *attention*, and users are *reminded* about the tracking limitations |
| 1.3 | When PII are used for direct marketing purposes, DS receives notification of possible objection. This notification may occur every 30 days | users *understand* that they can object to processing of their PII for direct marketing, and the limitations on those objections | - during registration, users must *opt-in* to processing for direct marketing or charitable purposes<br>- to ensure understanding and awareness, users are given examples and a *Just-In-Time Click-Through Agreement* (JITCTA) is used for final acceptance<br>- users are also *reminded* of their opt-in/out option in a preferences interface screen |
| 2 | Finality & Purpose Limitation: the use and retention of PII is bound to the purpose to which it was collected from the DS. | users *control* the use and storage of their PII | interface elements for making privacy decisions are prominent and *obvious* |
| 2.1 | The controller has legitimate grounds for processing the PII (see Principle 3.1) | users give implicit or explicit *consent* | click-through agreement should obtain *unambiguous consent* for controller to process the PII |

| | **Privacy Principle** | **HCI Requirement** | **Possible Solution** |
|---|---|---|---|
| 2.2 | Obligations: A controller must process according to his PS, controller also ensures other processors present a PS to be considered a recipient of the PII. When assessing a processor, the controller considers PII sensitivity and the similarity of processor PS to agreed-upon PS and location of the processor. The processor can only go beyond the agreed PS if: the processor's PS is state security, or prevention/detection/prosecution of criminal offences, or economic interests of the state, or protection of DS, or rights of other natural persons, or scientific/statistical analysis | users *understand* that their PII could be used for other purposes in special cases | - *user agreement* states that PII can (must) be passed on in special cases<br>- when viewing the processing logs, entries with limited information are coded to draw *attention*, and users are *reminded* about the special cases |
| 2.3 | Retention: the DS is to be presented a proposed retention period (RP) prior to giving consent, except where PS is scientific/ statistical. Controller ensures processor complies with RP, except where PS is scientific/statistical. When RP expires, it is preferably deleted or made anonymous. A record should be kept of processor's and controller's past adherence to RPs. | - users are *conscious* of RP prior to giving *consent*<br>- users are *aware* of what happens to their data when the retention time expires | - When data is provided, a retention period entry field will be *highlighted*<br>- Users are *informed* when information is deleted or made anonymous because of retention period expiry. |
| 3 | Legitimate Processing: Legitimate Processing (LP) is where the PII is processed within defined boundaries. | users *control* the boundaries in which their PII is processed | interface elements for making privacy decisions are prominent and *obvious* |
| 3.1 | Permission: To legitimately process PII, controller ensures that one or more of the following are true: the DS gives his explicit consent, the DS unambiguously requests a service requiring performance of a contract, the PS is legal obligation or public administration, or the vital interests of the DS are at stake. When matching the PS agreed to by the DS and the PS of the possible processor, any of the following will prevent processing: The controller/processor's actual PS differs from the PS consented to by the DS, the controller/processor intends passing the PII to a new processor, the controller/processor is not located in the EU, or the processor is violating a fundamental right to be left alone | - users give *informed consent* to all processing of data<br>- users *understand* when they are forming a contract for services, and the implications of that contract<br>- users *understand* the special cases when their data may be processed without a contract | - *JITCTA* to confirm unambiguous consent to data processing<br>- *JITCTA* to confirm the formation of a contract, and the implications/limitations of the contract<br>- in the tracking interface, include a *reminder* of special cases when data can be processed without a contract |
| 3.2 | Sensitive Data: The controller may not process any PII that is categorized as religion, philosophical beliefs, race, political opinions, health, sex life, trade union membership, or criminal convictions unless the DS has given their explicit consent or the processor is acting under a legal obligation | when dealing with highly sensitive information (religion, race, etc.), *users provide explicit, informed consent* prior to processing | if sensitive information is provided by the user, use a *double JITCTA* to obtain unambiguous consent for its processing |
| 4 | Rights: DS has the right to self-determination within the boundaries and balance of The Directive. | users *understand* and *can exercise* their rights | - at registration, use a *click-through agreement* to ensure that users know their rights<br>- interface layout provides *obvious* tools for controlling the rights functions |

| | **Privacy Principle** | **HCI Requirement** | **Possible Solution** |
|---|---|---|---|
| 4.1 | Access: DS is conscious of her rights. The DS has right to retrieve this data on PII processing: (1) who has received it; (2) who gave it to them; (3) when; (4) for what PS & (5) if a delete or anonymize operation has been acknowledged & authenticated. Items (1) (3) (4) should be disclosed if the proposed PS is any one of: state security, prevention/detection/prosecution of criminal offences, economic interests of the state, legal regulation, or protection of rights and freedoms (of other persons). If the DS is below the age of consent then access requests must be made by his/her legal representative (LR). In all cases, authentication should be proportional to the PII sensitivity | - users are *conscious* of their rights, which include right to know who has received their data, from whom, when, and why, and they *understand* the exceptions to these rights<br>- users *understand* and *can exercise* their rights | - the tracking functions are displayed *prominently*<br>- the exceptions to the rights are presented in the *user agreement*, and *reminders* are provided in the tracking interface |
| 4.2 | Control: DS may issue erase, block, rectify, or supplement commands on their PII. The DS is informed of the result of their command within 30 days. The communication is either: request accepted and executed, or request denied and an explanation. If the PII will not be editable due to the storage strategy applied, then DS is informed & asked to consent prior to providing any PII. Controller is accountable for the correct execution of DS requests for erase, block, rectify, or supplement the PII | - users are *conscious* of their rights, they can *exercise* control over their data, with ability to erase, block, rectify, or supplement the data<br>- users are *informed* when data will not be editable and they provide *consent* to processing | - the tracking functions are displayed *prominently*<br>- the exceptions to the rights are presented in the *user agreement*, and *reminders* are provided in the tracking interface<br>- the *commands* to erase, block, rectify, and supplement are associated with the tracking logs and *obvious* to operate<br>- a *JITCTA* is used when data will not be editable |
| 4.3 | Objections: if DS has not given direct consent to processing and the PS is public administrative or Legitimate Processing, the controller determines validity of the objection. If the PII is sensitive data and/or the PS is sensitive then the objection is accepted and the PII is deleted. If the PS is direct marketing then any objection is accepted and the PII is deleted. | users are *empowered* to object to processing for certain purposes | the tracking logs contain a *prominent function* to object to the processing |
| 4.4 | Derived Information: Certain PS supplied by processor to controller or controller to DS could be used to gain an insight into a person's personality, e.g., services of interest to the DS. This derived information shall not be processed unless: the DS is informed of the PS related to the derived information, he/she unambiguously requests a service requiring performance of a contract and has issued explicit consent. The DS can object to the processing of the derived information at any time, and the derived information must be deleted. | users *understand* and are *informed* that their behavior may provide some information, and they have provided *consent* for the processing of this information. They are also *empowered* to object to this processing | - the concept of derived information is *explained* at registration, and an example is provided<br>- a *JITCTA* is used to confirm consent to processing<br>- processing logs or other results of derived information are always presented with an *obvious* interface for objection |

# Thwarting Web Censorship with Untrusted Messenger Discovery

Nick Feamster, Magdalena Balazinska, Winston Wang,
Hari Balakrishnan, and David Karger

MIT Laboratory for Computer Science
200 Technology Square, Cambridge, MA 02139
{feamster,mbalazin,wwww,hari,karger}@lcs.mit.edu

**Abstract.** All existing anti-censorship systems for the Web rely on proxies to grant clients access to censored information. Therefore, they face the *proxy discovery problem*: how can clients discover the proxies without having the censor discover and block these proxies? To avoid widespread discovery and blocking, proxies must not be widely published and should be discovered in-band. In this paper, we present a proxy discovery mechanism called *keyspace hopping* that meets this goal. Similar in spirit to frequency hopping in wireless networks, keyspace hopping ensures that each client discovers only a small fraction of the total number of proxies. However, requiring clients to independently discover proxies from a large set makes it practically impossible to verify the trustworthiness of every proxy and creates the possibility of having untrusted proxies. To address this, we propose separating the proxy into two distinct components—the *messenger*, which the client discovers using keyspace hopping and which simply acts as a gateway to the Internet; and the *portal*, whose identity is widely-published and whose responsibility it is to interpret and serve the client's requests for censored content. We show how this separation, as well as in-band proxy discovery, can be applied to a variety of anti-censorship systems.

## 1 Introduction

Many political regimes and corporations actively restrict or monitor their employees' or citizens' access to information on the Web. Many systems try to circumvent these censorship efforts by using cooperative proxies. Anonymizer [1] is one of the oldest such systems. Peekabooty [15], Safeweb [11], and Zero Knowledge's WebSecure [13] use an SSL-encrypted channel to communicate requests to proxies outside of the censored domain, which then return the censored content over this encrypted channel. In Infranet [3], clients communicate with cooperating proxies by constructing a covert and confidential channel within an HTTP request and response stream, without engendering the suspicion that a visibly encrypted channel might raise.

These systems require a client within the censored domain to discover and communicate with a cooperating proxy outside of the domain, as shown in Figure 1. Each of these systems assumes that a censor blocks access to a Web server

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 125–140, 2003.

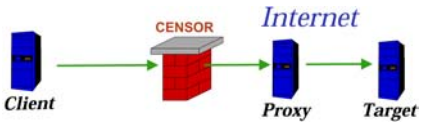



**Fig. 1.** Current censorship circumvention schemes rely on access to trusted proxies that serve clients' requests for censored content.

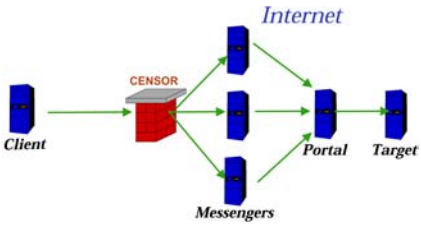



**Fig. 2.** Forwarding a message and decoding that request can be decomposed into two separate operations.

based on its identity (i.e., IP address or DNS name) and that the censor allows access to any host that does not appear to be delivering objectionable content. Thus, the livelihood of these systems depends on the existence of proxies that the censor does not know about.

All proxy-based censorship avoidance systems face the troubling *proxy discovery problem.* To gain access to censored content, clients must have access to cooperating proxies. However, if the censor can operate under the guise of a legitimate client, it can discover these proxies and block access to them. For example, China's firewall previously blocked access to the Safeweb proxy. An effective proxy discovery technique must allow a client to easily discover a few participating proxies but make it extremely difficult for a censor to discover *all* of these proxies. Any reasonable solution to the problem must defend against both *out-of-band* discovery techniques (e.g., actively scanning or watching traffic patterns) and *in-band* ones (e.g., where the censor itself becomes a client).

To achieve these goals, a proxy-based censorship avoidance system should have the following characteristics:

- *The system should have a large number of proxies.* A system with no more than a few proxies is useless once those proxies are blocked. A system with more proxies makes it more difficult for a censor to block all of them.
- *Clients must discover proxies independently of one another.* If every client discovers the same few proxies, a censor could block access to these popular proxies and render the system useless.
- *The client must incur some cost to discover a proxy.* Because the censor can assume the identity (i.e., IP address) of any client behind its firewall, it is relatively easy for a censor to operate a large number of clients solely to discover proxies. As such, discovering a proxy should require a non-trivial investment of resources, such as solving a client puzzle [6].
- *Brute-force scanning techniques must not expose proxies.* A censor may suspect that a host is a proxy and try to verify this in some fashion (e.g., by acting as a client and seeing if it acts as a proxy, etc.). Thus, to an arbitrary end-host, a proxy should look innocuous.

We propose a proxy discovery technique called *keyspace hopping* that limits in-band discovery of proxies by ensuring that no client knows more than a small random subset of the total set of proxies. The technique also prevents out-of-band discovery by distributing client requests across the set of proxies and ensuring that each cooperating end-host only assumes the role of a proxy for a small set of clients at any given time.

The requirement that clients discover proxies independently implies that clients will utilize arbitrary proxies that they may not trust. This introduces a fundamental tradeoff: while having a large number of independently discoverable proxies makes the system more robust to being blocked, it also makes it increasingly difficult to ensure that all proxies are trustworthy. An ideal proxy discovery system should be resistant to blocking and ensure that the client only exposes its requests for censored content to trusted parties.

We propose a solution that achieves this goal by recognizing that the proxy actually serves two functions: *providing access* to content outside the firewall, and *serving requests* for that content. Our solution, summarized in Figure 2, employs a large number of untrusted *messengers*, which carry information to and from the uncensored Internet, without understanding that information; and a smaller number of *portals*, which a client trusts to faithfully serve requests for censored content without exposing its identity.

## 2 Proxy Discovery Using Keyspace Hopping

Proxy-based anti-censorship systems must enable clients to discover proxies without enabling the censor to discover and block access to all of the proxies. Existing systems assume that there is some way to enable this discovery, but the problem has no obvious solution when the censor can become a client. Because of this possibility, *no single client (or small group of clients) should ever discover all proxies*. Proxies must come into existence more quickly than the censor can block them, and proxy discovery must be based on some client-specific property like IP address to raise the cost of impersonating many clients. In this section, we explore the design space for proxy discovery and describe our proposed mechanism, called *keyspace hopping*, that controls the rate at which any one client can discover proxies. In this section, we assume that the censor cannot operate a proxy, except for our analysis of in-band discovery in Section 2.3. We discuss how to completely relax this assumption in Section 3.

### 2.1 Design Considerations for Proxy Discovery

Anti-censorship systems should ensure that almost every client can always contact at least one proxy, even if the censor is able to block some of these proxies. The set of proxies should be difficult enough to discover that the only reasonable response by the censor would be to block access to the entire Internet.

A censor can discover proxies in two ways: *in-band*, by acting as a client of the anti-censorship system itself, and discovering proxies in the same manner as any

**Table 1.** A censor can discover and block proxies using either in-band or out-of-band discovery.

| Technique | Description | Design principles |
|---|---|---|
| *In-band* | Censor becomes a client and attempts to discover proxies in the same way a client would. | – Use client-specific properties for proxy discovery.<br>– Ensure no client can discover more than a small set of all proxies at any time. |
| *Out-of-band* | Censor uses traffic anomalies or active scanning techniques to discover proxies. | – Distribute clients evenly among available proxies.<br>– Ensure a host only acts as a proxy for a small subset of clients at any time. |

other client; and *out-of-band*, by actively scanning Internet hosts to determine whether any of them behaves like a proxy (we have previously explained the importance of maintaining proxy covertness for this reason [3]). Additionally, a censor can notice traffic anomalies that expose a proxy or a client, such as a sudden increase in traffic to a particular Web site or a group of clients that have very similar browsing patterns. Table 1 summarizes these discovery techniques and the corresponding design considerations.

**Limiting In-Band Discovery.** If we assume that a censor can become a client, the censor can use the same discovery mechanisms that a client uses to discover proxies. Thus, the set of proxies that any one client can discover should be small and relatively independent from the sets that other clients discover. This client-specificity implies that clients should discover proxies through some in-band mechanism (note that this is a departure from our previous thoughts about proxy discovery [3]).

To slow in-band discovery, we impose the following constraints: the proxies that any client discovers should be a function of some characteristic that is 1) reasonably specific to that client, 2) not easily modified, and 3) requires significant resources to compute. Two obvious characteristics of a client that satisfy the first two constraints are the client's IP address and subnet. Unfortunately, a censor that operates a firewall can easily assume an IP address or subnet behind that firewall. Hence, we must also require some significant investment of resources *per-client*, such as client puzzles [6], that makes it reasonably expensive for one entity to assume many different identities.

**Limiting Out-of-Band Discovery.** A censor might try to discover proxies using out-of-band discovery techniques. For example, all Web servers that run an Infranet responder might behave in a similar fashion (e.g., providing slower than normal Web response times, etc.). Alternatively, if many clients send requests to a single proxy within a small time period, a censor might notice a large increase in the number of connections to a host that does not ordinarily receive much traffic. It should be reasonably difficult for a censor to discover all proxies using these types of out-of-band discovery techniques.

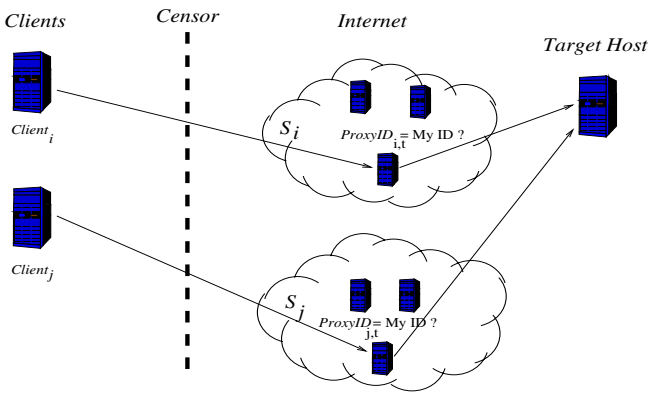



**Fig. 3.** In keyspace hopping, clients and proxies agree on which proxy forwards which client's request. Each client discovers a unique set of proxies.

To make out-of-band discovery more difficult, a host should only act as a proxy for a certain subset of clients at any time. This prevents one proxy from attracting traffic from an abnormally large number of clients. More importantly, it prevents a host from always appearing as a proxy to all clients, thus making it less likely that an out-of-band probe from an arbitrary host will expose the proxy. Furthermore, the set of clients that a proxy serves should change over time. This makes proxy discovery more difficult for the censor because the censor does not know which hosts are acting as proxies for which clients.

### 2.2 Keyspace Hopping

We apply the design principles from Section 2.1 to our proxy discovery system, called *keyspace hopping* because of its similarities to frequency hopping [9]. Frequency hopping is used in wireless communication; the basic idea is to modulate a signal on a carrier frequency that changes pseudorandomly over time. Wireless communication uses frequency hopping to resist jamming, since an adversary must either saturate the entire frequency band with noise or track the frequency hopper's choice of carriers.

We propose a similar idea, with the exception that the censor is attempting to jam communication channels by preventing a client from reaching any proxies. At any given time, a certain proxy (or set of proxies) agrees to serve requests for a client, and the client forwards its requests to that proxy, as shown in Figure 3. To block a client's communication with its proxies, the censor must block communication with all of the client's proxies.

Keyspace hopping must solve several problems. The first problem is *proxy assignment*: what is the appropriate mechanism for assigning clients to proxies? Next, clients must perform *lookup*: how do clients discover the IP addresses of their proxies while preventing the censor from performing arbitrary lookups to discover all proxies? Finally, the system must have a *bootstrapping* phase: how

can the client, initially knowing nothing, obtain the necessary information to discover its set of proxies? The rest of this section addresses these problems.

**Proxy Assignment.** To guarantee that no single client can ever discover a large fraction of the proxies, keyspace hopping assigns a small subset of all proxies to each client. To prevent proxies from being actively scanned, and to balance client requests across proxies, keyspace hopping dictates when a client can use a particular proxy in its subset.

To facilitate the mapping, each proxy is assigned a globally unique identifier *ProxyID*, such as a hash of the proxy's IP address. The set of proxies for a client is then determined by computing a client-specific index into the total keyspace, and by selecting a constant number of proxies whose identifiers most closely follow the index. The index is computed from a client-specific identifier (which could be, for example, the client's IP address) and a shared secret *hkey* that prevents an adversary from computing the subspace.

A client determines the proxy with which it communicates by adding the output of a uniform collision-resistant hash function, $B_i$ (its base index in the keyspace), to a time-dependent *PreProxyID*, which is determined from the output of a universally-agreed upon pseudorandom number generator, $\mathcal{G}$. *ProxyID* must be based on *hkey* to prevent a censor from recomputing a suspected client's sequence of *ProxyID*s and tracking a suspected client's path through a sequence of proxies (this is particularly important for Infranet, which seeks to preserve client deniability).

The following equations present the assignment more formally:

$$\begin{aligned} B_i &\leftarrow \mathcal{H}(\text{Client ID}, hkey) \\ PreProxyID_{t,i} &\leftarrow \mathcal{G}(\text{Client ID}, hkey, t) \\ ProxyID_{t,i} &\leftarrow (B_i + (PreProxyID_{t,i} \mod |S_i|)) \mod |P| \end{aligned}$$

where $S_i$ is the set of proxies that client $i$ knows about, and $P$ is the set of all proxies in the system[1]. Both $|S_i|$ and $|P|$ are well-known constants, and $|S_i|$ is the same for all clients $i$. $ProxyID_{t,i}$ is rounded up to the closest *ProxyID* in the client's set.

The size of the subset of the keyspace that client $i$ uses, $|S_i|$, addresses a fundamental design tradeoff—the flexibility gained through using more proxies vs. independence from the fate of other clients (obtained by not sharing proxies with other clients). Smaller proxy subsets decrease the likelihood that one client's proxies will share a proxy with some other client, but mean that a client may appear more conspicuous by sending more traffic to a smaller number of hosts. A client with smaller set of proxies is also less resilient to having proxies blocked.

To minimize the likelihood that the censor discovers a proxy and blocks it, we require that any proxy only serve a small subset of clients at any time. For a given request, the proxy must determine whether or not it should serve that

[1] We assume for simplicity that this value is constant. We describe how to relax this assumption in Section 2.3.

particular client's request for censored content. This is easily done—the proxy can simply determine the IP address from which the client request originated and check whether the computed *ProxyID* for the current time interval $t$ matches its own *ProxyID*. The proxy only treats the client's request as a request for censored content if this value matches (regardless of whether or not it is such a request) .

The client may need to rediscover proxies when the proxies that it is using either leave the system or become unreachable; this can be done in the same way that the client bootstraps its original set of proxies, as described below.

**Lookup and Initialization.** To use keyspace hopping to contact proxies, the client must know two things: the mapping from *ProxyID*s to IP addresses for its set of proxies, and the value for *hkey*.

To prevent a censor from harvesting IP addresses to discover and block proxies by performing arbitrary lookups, a client should only be able to lookup IP addresses for *ProxyID*s that it will use for keyspace hopping. The first proxy that a client contacts will return the *ProxyID* to IP address mappings for *only the ProxyIDs that the client needs to know*. Because the proxy that the client originally contacts knows the client's IP address and the value of *hkey* that it assigned to that client, the proxy also knows the set of proxies that the client will attempt to contact. To make it more difficult for a censor to assume the identities of many clients (e.g., by changing IP addresses, etc.), the proxy can make this discovery more expensive for the client by encrypting the mapping and forcing the client to solve a puzzle to recover the mapping, such as a partial brute force decryption given only $k$ bits of an $n$-bit encryption key [6]. To ensure that the client cannot bias its keyspace assignment, *hkey* must be set by a proxy, rather than chosen by the client. The proxy that assigns *hkey* must also inform the other proxies in the client's proxy set; if proxies are not trusted, this must be done differently, as discussed in Section 2.3.

**Bootstrapping.** With the approach we have described above, the client must contact some proxy that will send it *hkey* and *ProxyID* mappings. However, to perform this bootstrapping operation, the client must first know the IP address of at least one proxy. This sounds like the original problem: if the censor happens to block a well-known bootstrapping proxy, new clients will not be able to discover the subset of proxies that they need for keyspace hopping. Thus, we must also ensure that clients discover their first proxy reasonably independently.

A client only needs to discover *one* proxy to bootstrap. A client might already know of an operational proxy (out-of-band), or clients could establish a web of trust (as in PGP [8]) where a client could divulge one of the proxies in its subset to trusted friends. To prevent out-of-band discovery, a proxy should bootstrap a client that it has never seen before only with a certain probability. Alternatively, a proxy might bootstrap only clients that are referred to it explicitly by clients that it already knows.

### 2.3 Analysis and Discussion

In this section, we analyze how well keyspace hopping resists discovery and blocking. We also discuss the deniability properties of keyspace hopping.

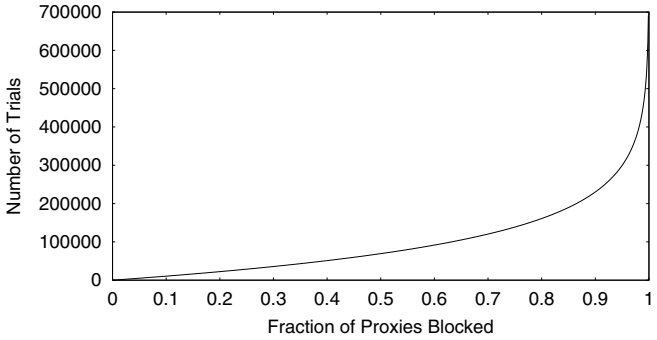



**Fig. 4.** Blocking 95% of $10^5$ proxies would requires the censor to solve about 300,000 puzzles.

**In-Band Discovery.** We analyze the likelihood that a given client will be denied access to any proxy in its subset of known proxies, given that a censor has the capability to impersonate a certain number of clients in a reasonable amount of time. In the bootstrapping phase, each client discovers a specific set of proxies based on some client-specific identifier (e.g., its IP address). Since the censor controls all of the IP address space behind the censorship firewall, it can impersonate any IP address behind its firewall to discover what set of proxies a client from that IP address might use.

Because a client cannot discover the proxies in its subset before solving a puzzle, a censor must solve one of these puzzles for each subset of proxies that it wants to discover. However, because each legitimate client will only have to solve the puzzle once, the puzzle can be sufficiently difficult (a draconian approach would require each client to spend a week decrypting its subset of proxies).

How many clients does the censor need to impersonate to know about a significant fraction of all proxies? Let's assume for simplicity that each puzzle allows the censor to discover one proxy selected randomly from $P$ (i.e., $|S_i| = 1$). If the censor already knows about $n$ proxies, then the probability of discovering a new proxy is $(P - n)/P$. Thus, assuming an independent Bernoulli process, the censor will discover a new proxy after impersonating $P/(P - n)$ clients, or $1/(P - n)$ of the total number of proxies. On average, the censor will have discovered $N$ proxies after $\sum_{k=1}^{N} P/(P-k)$ impersonations (this is known as the "coupon collector problem"). For example, if $P = 10^5$, then a censor must solve about 70,000 puzzles to discover 50% of the proxies, and about 300,000 puzzles to discover 95% of all proxies[2]. Figure 4 shows the relationship for $P = 10^5$—it becomes increasingly hard for the censor to discover and block all proxies, or even a large fraction of them. If we design the system so that it is difficult enough to solve each puzzle (e.g., a day per puzzle), then it will take the censor almost 200 years to discover half of the proxies. If the system is able to detect

[2] Recent studies suggest that the number of Web servers on the Internet is on the order of $10^7$ and growing [7]; having 1% of these act as proxies is a reasonable goal.

that it is being scanned by a censor, it can also increase the difficulty of the client puzzles to slow the censor down even further.

If a censor can operate a proxy, it can discover clients by determining which clients make requests for censored content. This problem arises because the proxy can identify clients solely based on which hosts are contacting it with meaningful requests. To address this, the proxy functionality can be decomposed into an untrusted *messenger* and a trusted *portal*, where only trusted portals should be able to determine which hosts request censored content (we describe this approach in detail in Section 3). In this case, the censor can operate a malicious messenger, but that messenger will not be able to distinguish anti-censorship requests from innocent messages; this is particularly true in the case of Infranet, where innocent clients will be sending HTTP requests to that messenger under normal circumstances. For this technique to be effective with other anti-censorship systems, there must be an innocent reason for a client to send messages to that messenger; otherwise, there is no plausible deniability for sending messages through that messenger.

The requirement that the proxy know *ProxyID* to IP address mappings and IP address to *hkey* mappings is problematic because this implies that the censor can discover other proxies by becoming a proxy and can discover clients by discovering *hkey* mappings. Of course, proxies can inject a large number of false IP address to *hkey* mappings; however, a better solution uses untrusted messengers and trusted portals to control who knows this information. Trusted portals can assign *ProxyID* to IP address mappings to clients. In this case, portals can inform messengers about which clients it should serve during any time slot without requiring messengers to ever learn of other messengers. Portals can also tell messengers about only the *hkey* mappings for clients that have that messenger in its set. Portals can also simplify proxy assignment, since they can inform clients about changing values of $|P|$ or simply compute the keyspace subset $S_i$ for each client $i$, using the current value of $|P|$. We discuss messengers in further detail in Section 3.

**Out-of-Band Discovery.** A censor can discover clients out-of-band by watching for traffic anomalies (e.g., a client sending messages to a specific set of hosts outside the firewall) and can discover proxies out-of-band by probing for behavior typical of a proxy (e.g., serving visible HTTP requests more slowly than a normal Web server, in the case of Infranet). Keyspace hopping makes out-of-band discovery more difficult because, given an arbitrary message from the censor, the proxy will ignore the censor's request.

A censor could mount out-of-band discovery by computing the sequence of proxies that a client would use to serve its requests and determining whether any clients send messages to proxies according to the same schedule. For this reason, the *ProxyID* for a particular client and time interval must depend not only on the client's IP address, but also some key *hkey* that is known only to the client and its set of proxies $S_i$. Thus, if the censor does not know *hkey*, it does not know either the keyspace for that client, nor does it know the progression of proxies that the client will take through that keyspace.

In the case where the censor operates at least one of the proxies in $S_i$, the censor knows all the information to hypothesize that a certain host might be operating an anti-censorship client. A simple solution relaxes frequency hopping to allow the client to pass requests to any of the *ProxyID*s that were valid for the $n$ most recent time intervals. However, this still allows the censor to ascertain that a suspected client is contacting machines that are within the client's proxy set $S_i$. Another solution is to distribute a set of *hkey*s to the client and allow the client to send messages on any one of multiple channels. The censor would then have to know the secrets for all of these channels to successfully track the client through a series of proxies.

**Deniability.** Infranet explicitly tries to make a client's requests for censored content as similar as possible to normal-looking Web traffic; we would like to preserve such deniability when incorporating keyspace hopping. Other anti-censorship systems do not provide deniability at all, so there is no risk of compromising client deniability in these cases.

Keyspace hopping presents several potential vulnerabilities that might compromise the client deniability goals of anti-censorship systems such as Infranet[3]. First, a client may arouse suspicion by attempting to contact a recently-blocked proxy. However, this weakness is no worse than that which exists in the original Infranet design. Second, the hopping sequence between proxies must be chosen so that both the hopping interval and the proxies between the client hops seems like a reasonable browsing pattern for a normal user. Because the keyspace hopping schedule we have presented does not rely on client-specific time intervals, a censor could potentially single out anti-censorship clients by correlating browsing request patterns with other known anti-censorship clients. Designing a hopping schedule that does not arouse suspicion is a challenging problem for future work.

Infranet clients should make visible requests to proxies in a way that preserves client deniability. Keyspace hopping does not affect this aspect of client deniability since it only affects how a client hops between proxies (i.e., between "responders", in Infranet parlance), not the visible requests the client makes to any particular responder.

## 3 Communication through Untrusted Messengers

With the keyspace hopping technique that we described in Section 2, the client cannot verify the trustworthiness of every proxy that it contacts. In this section, we describe how to rectify this problem. Specifically, we decompose the functions of the proxy into two distinct modules: the *messenger*, which acts as the gateway, or access point, to the Internet, and the *portal*, which deciphers and serves clients' requests, as shown in Figure 2. The *messenger* acts as untrusted intermediary through which the client and portal communicate[3].

[3] This differs from the Triangle Boy approach, where the messengers (i.e., Triangle Boy nodes) must be (1) widely announced and (2) trusted (since they are intermediaries in the SSL handshake with the Safeweb server).

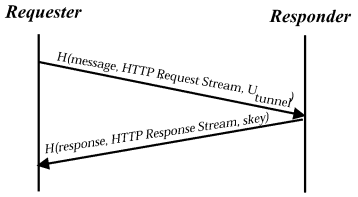



**Fig. 5.** Without messengers (original design).

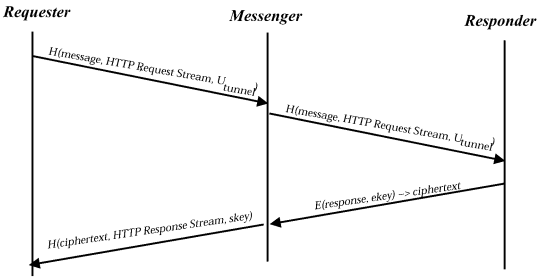



**Fig. 6.** With messengers.

Because traffic passes through the messenger in the same way that it passes through the censor, a messenger can mount every attack that a censor can mount (as outlined in previous work [3]); in addition, the messenger can disrupt communication between the client and the proxy by failing to deliver packets to the intended destination. In this section, we describe how an untrusted messenger can be implemented in the context of both Infranet and SSL-based systems.

### 3.1 Infranet Messenger

The original Infranet design proposes that clients circumvent censors by sending requests via an Infranet *requester*, which hides requests for censored content in a visible HTTP requests to the Web site of the Infranet *responder*, as shown in Figure 5. In the case of Infranet, the responder acts as the portal. To separate forwarding messages from decoding messages and serving requests, we use two separate entities: the Infranet *messenger*, which is the machine that the client directly communicates with; and the Infranet *responder*, which uses the same upstream and downstream modulation techniques as before. The messenger simply acts as a conduit for the requester and responder's messages. We present an overview of the necessary modifications to the original Infranet requester/responder protocol and follow with a discussion on the implications of the Infranet messenger on deniability and other security properties of Infranet [3].

**Overview.** Figure 7 shows the Infranet architecture with the separation of the Infranet proxy into a messenger and responder (i.e., portal). The Infranet responder functions as before and assumes responsibility for translating the Infranet requester's visible HTTP requests into requests for censored content. The messenger informs the responder about the visible HTTP requests of certain clients (e.g., from Section 2, those which should be hopping to its *ProxyID*), and hides the appropriate encrypted content in its HTTP responses for the appropriate users. Figures 5 and 6 show the conceptual distinction between the two versions of the Infranet protocol with and without the messenger. For simplicity, we discuss the comparison for steady-state communication only.

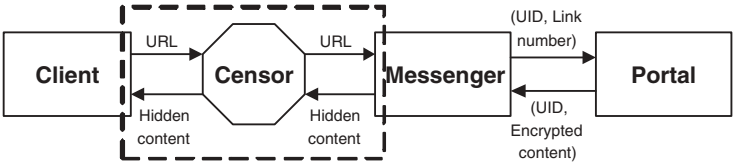



**Fig. 7.** An improved architecture separates the forwarding and decoding of hidden messages in both directions. This allows a potentially untrusted messenger to service requests and serve hidden content. The UID serves to demultiplex requesters.

Without Infranet messengers, the requester hides a message using the hiding function $\mathcal{H}$ and an upstream modulation function $\mathcal{U}_{tunnel}$ known only to the requester and responder. In the downstream direction, the responder hides the requested content with the downstream hiding function (e.g., steganography), using a secret hiding key *skey* known only to the requester and responder. Using untrusted messengers does not affect upstream hiding; the messenger simply tells the responder which request was made by a particular requester, but the message remains hidden, as far as the messenger is concerned. The output of the upstream hiding function is an HTTP request stream, and it suffices for the messenger to pass this request sequence directly to the responder. Note that this is an HTTP request stream for objects on the *messenger's* Web site, which the responder can then decode into message fragments (as described in previous work [3]). The responder no longer needs to run a Web site, although the messenger must do so. Only the requester and responder understand the semantics of the visible HTTP request stream.

The downstream communication protocol is similar to that proposed in the original Infranet design, except that *two* keys must be used in the downstream direction. In the original design, the Infranet responder encrypts and steganographically embeds the requested content in images from its Web site that it would subsequently serve to the client. In this case, the responder can use the same key to encrypt and steganographically embed the content[4]. However, because the messenger is not necessarily a trusted entity (i.e., it could in fact be a malicious node), the responder must first separately encrypt the requested content *under a key that is unknown to the messenger*. However, the messenger must steganographically embed the requested content in its own visible HTTP responses, and thus needs a separate key to do so. Of course, the requester must know both of these keys to successfully retrieve the hidden content; the responder can generate these keys and send them to the requester as in the original Infranet design. Thus, the three parties must now come to agreement on two keys: an **encryption key**, *ekey*, that is shared between the requester and responder, and is unknown to the messenger; and a **hiding key**, *skey*, which the requester and the messenger must know, and the responder may also know.

[4] Technically, encryption of the content to be hidden is a part of the steganographic embedding [10], but we mention both operations separately for clarity.

**Analysis and Discussion.** Infranet provides client deniability disguising client request stream as a user's "normal" browsing pattern because the requester's browsing pattern is determined in the same manner as before by an upstream modulation function as agreed upon by the requester and responder. Because introduction of a messenger does not affect the requester's use of an upstream modulation function, the stream of visible of HTTP requests and responses still looks innocuous to any entity that does not know the upstream modulation function. Solely based on seeing the HTTP request stream from a client, the messenger has no more knowledge about whether a client is an Infranet requester or an innocent client; only the responder knows how to map this request stream to the requester's hidden message.

In the downstream direction, the messenger knows that it is embedding ciphertext in one of its images and returning that ciphertext to some client. However, it does *not* know 1.) whether that ciphertext contains any useful data or what that data might be, or 2.) if that ciphertext corresponds to a request made by a particular client. A responder could return bogus content for clients that are not Infranet requesters without the messenger's knowledge.

Because we have separated the process of forwarding messages from decoding these messages, a requester does not need to trust the messengers, but it still needs to trust the responder. This separation allows the identity of responders to be widely published, since a censor's knowledge about the identities of Infranet responders does not enable it to block a client's access to the messengers. Thus, requesters can pass messages through a set of untrusted messengers (which, as we know from Section 2, can be made resistant to complete discovery and blockage) to well-known, trusted Infranet responders.

While a malicious messenger cannot distinguish clients that are making requests for censored content from ordinary Web clients (since only Infranet requesters and responders know whether the visible HTTP request stream has any hidden semantics), it can certainly disrupt the communication between the requester and responder by refusing to pass some messages or message fragments from the requester to the responder, and vice versa. For example, the messenger may fail to pass some URLs that it hears from a requester along to the responder; alternatively, it might neglect to embed certain pieces of content in responses to the requester. These are the same types of attacks that the censor can perform itself at the firewall; previous work provides detailed discussion about how to handle these types of attacks [3]. The client can easily detect these types of attacks—either the responder will serve an incomplete or wrong request, or the requester will not receive the full data that it requested. Presumably, this messenger could then be marked as malicious, faulty, or misbehaving, and removed from the set of candidate messengers.

### 3.2 Messengers for SSL-Based Systems

Other existing systems, including Safeweb/Triangle Boy [11], Zero Knowledge's WebSecure [13], and Peekabooty [15], use an encrypted channel between the client and the proxy to send requests and receive censored content. Although

these systems do not provide *covert* censorship circumvention (SSL is vulnerable to fingerprinting attacks, for one [5,12]), these systems nevertheless potentially allow clients to circumvent censorship techniques using one or more proxies. Nevertheless, these SSL-based proxy systems can also benefit by separating the proxy into a messenger and a portal, which would allow them to use the messenger discovery techniques described in Section 2.

It might seem that we could use a messenger as a conduit for an SSL connection in the same way that was possible for the Infranet messenger. In fact, SSL-based proxies are less amenable to the separation of the proxy into a messenger and a portal—traffic must appear to originate from the messenger, but the SSL handshake includes a step whereby the portal returns to the client a certificate with the portal's public key. Using this naive approach, these systems cannot attain the same level of resistance to blocking that a system that is not based on SSL can achieve. Any modifications to the SSL protocol itself (e.g., removing this portion of the handshake, etc.) would also arouse suspicion from the censor, which we would like to avoid.

Using onion-routing to tunnel the initial SSL handshake results in connection establishment that does not require suspicious modifications to SSL and is more robust to the presence of untrusted messengers [14]. For example, with knowledge of a messenger's public key, a client can encrypt its half of the SSL handshake with the messenger's public key, and the messenger can unwrap this and send it to the portal. The messenger must also establish the equivalent of a reply block, so that the messenger can send the portal's half of the SSL handshake encrypted back to the client. Using the discovery mechanisms proposed in Section 2, however, it is not possible for the client to trust the messenger's public key. To achieve greater assurance that these untrusted messengers will not compromise the client's confidentiality, the client can specify that the initial handshake be routed through multiple messengers. An alternative approach would be to use Tarzan [4] to establish the initial SSL handshake, or even to conduct the entire communication over Tarzan.

## 4 Conclusion

We have presented the *proxy discovery problem*, which is faced by every proxy-based anti-censorship system: how can clients discover the proxies that will assist them in gaining access to censored information without having the censor discover and block these proxies? Because a censor can discover proxies both in-band (by becoming a client itself) and out-of-band (by actively scanning for proxies, or by noticing odd traffic patterns between clients and suspected proxies), our techniques ensure that it is difficult for a censor to discover more than a small subset of all proxies using either method. We have proposed *keyspace hopping*, which defends against both in-band and out-of-band widespread discovery by any one client. Because each client selects its proxy from a set determined by client-specific information that is not easily forged (i.e., the client's IP network), it is difficult for any one client to discover a large set of proxies. In addition,

proxies are configured to "hop" with clients, so each one will only act as a proxy for some small subset of clients at any given time.

Because keyspace hopping does not allow clients to choose specific proxies, clients must use untrusted hosts as gateways to the uncensored Internet. To remedy this problem, we have separated the functions of the proxy into two distinct components—an untrusted *messenger*, which clients discover through keyspace hopping and only serve to pass along clients' hidden messages to *portals*, widely-known and trusted hosts with which clients communicate to request and retrieve censored content. Although messengers have the ability to disrupt communication between clients and portals, messengers cannot distinguish anti-censorship clients from innocuous clients[5]. This separation also allows the identities and public keys of portals to be widely-published, since knowledge of these hosts does not allow a censor to block access to messengers.

This paper presents many possibilities for future work. We intend to develop a prototype of our proposed designs for use with Infranet. Designing a keyspace hopping sequence that more closely mimics the habits of normal browsing remains an open question. The proxy discovery problem mirrors the structure of "leaderless resistance" social networks, which are composed of small, independently-operating sets and are robust to infiltration by disruptive agents [2]; we may gain insight into the proxy discovery problem by studying the structure of these networks more closely.

## Acknowledgments

We are grateful to David Andersen for many helpful discussions and for the suggestion of using client puzzles. Thanks also to Jean Camp and Daniel Rubenstein for thoughtful discussions, and to Sameer Ajmani, Kevin Fu, Stuart Schechter, and the anonymous reviewers for comments on drafts of this paper.

## References

1. Anonymizer. `http://www.anonymizer.com/`.
2. Louis Beam. Leaderless resistance. `http://www.louisbeam.com/leaderless.htm`, February 1992.
3. Nick Feamster, Magdalena Balazinska, Greg Harfst, Hari Balakrishnan, and David Karger. Infranet: Circumventing Web censorship and surveillance. In *Proceedings of the 11th USENIX Security Symposium*, San Francisco, CA, August 2002.
4. Michael J. Freedman and Robert Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, Washington, D.C., November 2002.
5. A. Hintz. Fingerprinting websites using traffic analysis. In *Workshop on Privacy Enhancing Technologies*, San Francisco, CA, April 2002.

[5] In Infranet, an innocuous client is an ordinary Web client. For SSL-based schemes, an innocuous client would be an onion-routing or Tarzan client.

6. A. Juels and J. Brainard. Client puzzles: A cryptographic defense against connection depletion attacks. In *Proceddings of the Network and Distributed System Security Symposium (NDSS'99)*, San Diego, CA, February 1999.
7. Netcraft web server survey. `http://www.netcraft.com/survey/`, 2003.
8. PGP FAQ. `http://www.faqs.org/faqs/pgp-faq/`.
9. J. Proakis and M. Salehi. *Communication System Engineering*. Prentice-Hall, Englewood Cliffs, NJ, 1994.
10. N. Provos. Defending against statistical steganalysis. In *Proceedings of the 10th USENIX Security Symposium*, Washington, D.C., August 2001.
11. SafeWeb. `http://www.safeweb.com/`.
12. Qixiang Sun, Daniel R. Simon, Yi-Min Wang, Wilf Russell, Venkat Padmanabhan, and Lili Qiu. Statistical identification of encrypted Web browsing traffic. In *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, May 2002.
13. Zero-Knowledge Systems. Freedom WebSecure. `http://www.freedom.net/products/websecure/`.
14. Paul F. Syverson, David M. Goldschlag, and Michael G. Reed. Anonymous connections and onion routing. In *Proceedings of the 18th Annual Symposium on Security and Privacy*, Oakland, CA, May 1997.
15. The Cult of the Dead Cow (cDc). Peekabooty. `http://www.vnunet.com/News/1121286`.

# GAP – Practical Anonymous Networking*

Krista Bennett and Christian Grothoff

S$^3$ lab and CERIAS,
Department of Computer Sciences, Purdue University
kbennett@cerias.purdue.edu, grothoff@cs.purdue.edu
http://www.gnu.org/software/GNUnet/

**Abstract.** This paper describes how anonymity is achieved in GNUnet, a framework for anonymous distributed and secure networking.
The main focus of this work is GAP, a simple protocol for anonymous transfer of data which can achieve better anonymity guarantees than many traditional indirection schemes and is additionally more efficient. GAP is based on a new perspective on how to achieve anonymity. Based on this new perspective it is possible to relax the requirements stated in traditional indirection schemes, allowing individual nodes to balance anonymity with efficiency according to their specific needs.

## 1 Introduction

In this paper, we present the anonymity aspect of GNUnet, a framework for secure peer-to-peer networking. The GNUnet framework provides peer discovery, link encryption and message-batching. At present, GNUnet's primary application is anonymous file-sharing. The anonymous file-sharing application uses a content encoding scheme that breaks files into 1k blocks as described in [1]. The 1k blocks are transmitted using GNUnet's anonymity protocol, GAP. This paper describes GAP and how it attempts to achieve privacy and scalability in an environment with malicious peers and actively participating adversaries.

The GNUnet core API offers node discovery, authentication and encryption services. All communication between nodes in the network is confidential; no host outside the network can observe the actual contents of the data that flows through the network. Even the type of the data cannot be observed, as all packets are padded to have identical size. Availability is guarded by an accounting scheme that is based upon link authentication and which does not require end-to-end knowledge about transactions [10].

The goals of the GNUnet project are to explore the possibilities and limitations of secure peer-to-peer networking. Achieving privacy is a significant problem for peer-to-peer technology, even when no spyware is bundled with applications. Additional security features are needed before peer-to-peer networks can be trusted to store more sensitive data, such as medical records. GNUnet is a strict peer-to-peer network in which there are no nodes exercising control over the network.

* Portions of this work were supported by sponsors of CERIAS.

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 141–160, 2003.

Any kind of central server would open the network to attacks, whether by attackers trying to control these entities, legal challenges, or other threats which might force operators of such critical and exposed nodes out of business. The best way to guard against such attacks is not to have any centralized services.

GAP strives to achieve initiator and responder anonymity in relation to all other entities, including GNUnet routers, active and passive adversaries, and the responder or initiator respectively. The actions involved in publishing content are indistinguishable from those involved in responding to requests; thus, responder anonymity covers publisher anonymity in GAP. It is not possible for peers to retrieve content from publishers that do not use GAP. Also, content migrates over the network. Because of this, even if responder anonymity is broken, there will be no certainty that the original publisher has been identified. While anonymity for intermediaries would be desirable, participation in a protocol on the Internet is generally visible to any powerful adversary. Thus, GAP does not strive to hide participation in the protocol. For GAP, it is only important that no adversary can correlate an action with the *initiating* participant.

The most significant difference between GAP and prior mix-based protocols is that traditional mix protocols always perform source rewriting at each hop. GAP mixes can specify a return-to address other than their own, thereby allowing the network to route replies more efficiently. GAP does not attempt to avoid a direct network connection between initiator and the responder. In order to achieve anonymity, it is only important to decouple the relationship between the initiator and the action. Thus, anonymity is achieved if an adversary cannot determine the initiator of an action. This can be achieved by making the initiator look like an intermediary: a participant that is merely routing data. This realization allows GAP to bypass a typical restriction on most indirection-based anonymous routing protocols which require that either the reply takes exactly the same path as the request [5,13] or the path is statically predetermined and cannot be optimized en route [3,12,21]. Some protocols, like [19], use multicasts for the reply, but these consume large amounts of bandwidth.

In order to understand how GAP works it is important to realize that given a powerful global passive adversary, the operation of proxy services like the website `anonymizer.com`, which are generally perceived to be anonymizing requests from their customers, degenerates to a situation where the customers merely provide cover traffic for the proxy service. The only entity which can then proceed with reasonable anonymity by using the proxy service is the actual operator of the proxy service (if they use the service from within). Similar problems arise for many other centralized anonymity systems (even the ones that use distributed trust), since they do not provide cover traffic for senders and receivers. A global passive adversary can attack these schemes by treating the whole set of mixes as a black box and simply looking at the messages going in and coming out at the initial entry point and final exit point. This type of network-edge analysis is made impossible in a peer-to-peer network where receivers and senders are part of the mix. Tarzan [9] is an anonymizing peer-to-peer infrastructure where the initiators are part of the mix network. Tarzan cannot anonymize responders since

they are not part of the mix network. Since censors typically prefer to attack the small set of publishers instead of the large group of readers, GAP's approach of anonymizing both senders and receivers has potentially broader applicability and greater usefulness in censorship-resistant networking.

### Anonymity and the Adversarial Model

A communication is defined to be anonymous with a probability $p$ if the adversary cannot prove with probability greater than $p$ that a node was the initiator or the responder in that communication. For the discussion in this paper, the adversary is assumed to have no means other than the interactions of the nodes via the protocol to determine if a node was the initiator. The paper will show that given certain assertions about the adversary, a node using GAP can determine its degree of anonymity and trade anonymity for efficiency. Note that the question of whether or not a system is *sufficiently* anonymous for a specific application depends entirely on that application's purpose and, thus, the choice of $p$ may vary. This allows nodes to operate more efficiently whenever applicable.

It is important to note that the **burden of proof** is put on the adversary. The adversary must identify a node and prove that a communication originated from there. If the legal system were to require the *initiator* to disprove being the origin, anonymity becomes essentially illegal. Aside from the restriction of only using evidence obtained from protocol-related interactions to prove that a node performed a particular action, the adversarial model for GAP allows the adversary to do almost anything but break cryptographic primitives. The adversary is assumed to see all encrypted and unencrypted traffic between all nodes at all times, but cannot decrypt encrypted traffic between two nodes where neither node is controlled by the adversary. The adversary controls an arbitrary number of nodes in the network and the nodes are free to collaborate out-of-band. Adversarial nodes can violate the protocol and/or be well-behaved participants in the network. The adversary can also interrupt communications between arbitrary nodes in the network. Since GNUnet is a peer-to-peer network in which every peer has willingly joined and where no messages are sent to machines that are not part of the overlay network, problems with exit-nodes [6] which communicate with the final recipient (who often is not part of the network) do not arise.

While the adversary described above is extremely powerful, its power must be limited slightly. The reason for this is that if the adversary is able to control (or at least decrypt) all traffic that a node sends or receives, the node cannot engage in any anonymous communication. This is true for all anonymous networks, however. Thus, in order to provide the user with any degree $p > 0$ of anonymity, any protocol requires that a node must be able to communicate with at least one other node that is not controlled by a malicious adversary.

GAP cannot prove that the adversary is bound by this constraint in practice. In fact, even a powerful adversary that obeys this rule can break anonymity in GAP with a certain probability $p$. The degree of anonymity $p$ provided by GAP depends on the strength of the adversary and the internal trade-offs that the

individual node chooses to take. Note that the trade-off is not negotiated with other peers and does not impact their operation. The determination of the degree of anonymity that can be achieved is based on an estimate of the power of the adversary, which must essentially be guessed. If the guess is that the adversary controls all communications, the result would be that in order to achieve any degree $p > 0$ of anonymity, no communication may take place.

## 2 Anonymity in GNUnet

This section describes how anonymity is achieved in GNUnet using GAP. In order to be able to evaluate the anonymity guarantees that GAP provides, a new perspective on how indirecting communications results in anonymity is first described. Then the scheme employed in GAP is laid out and its guarantees and efficiency are discussed.

### 2.1 Hiding the Initiator of Activity

Consider the scenario illustrated in figure 1. In this scenario, a node receives two queries and sends three. In this picture, the two nodes that sent their queries

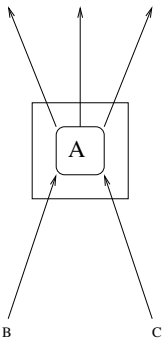



**Fig. 1.** Hiding

are exposed; node $A$ can correlate these nodes with their queries, as a traffic analysis reveals that both $B$ and $C$ sent a query. Also an external adversary can tell that $B$ and $C$ started some communication. If $A$ is allowed to send a query twice, traffic analysis alone cannot reveal if $A$ sent a new query or was merely indirecting queries from other nodes.

In this sense, indirections do not hide the original senders $B$ and $C$ from powerful adversaries; instead, indirections obfuscate what the node that is indirecting is doing. No scheme that tries to achieve anonymity on an observable, open network can hide the fact that a node is *participating*. The best a scheme can do is guarantee that no adversary can distinguish activity that a node initiates from mere participation in the protocol. The example above demonstrates that a node can hide its own activities by handling traffic for other nodes.

### 2.2 Protocol Overview

The GAP protocol consists of two types of messages: queries and replies. A query consists of a resource identifier (in GNUnet, RIPE160 hash codes are used) and a node identifier that describes where the reply should be sent. This reply field is the primary difference of the wire-format compared to protocols such as Freenet or Crowds where the reply always goes to the sender of the query. A time-to-live field is also included for routing purposes. The time-to-live has a pseudo-random initial value and is decremented by routers with additional pseudo-random components in the expression. A reply is merely the data that was requested. Communication between nodes uses link encryption; each node is linked to as many nodes as possible.

The resource identifier of a query is passed to GAP by the application layer. If a reply is not received after a certain (randomized and exponentially increasing) amount of time, the query is retransmitted. Queries are searches; success is not guaranteed. For example, the resource may simply be unavailable at the time of the query. The application layer is responsible for deciding when to give up.

After a node receives a query, it processes the query by taking the following steps:

1. Determine if the node is too busy to process the query. This check includes CPU and bandwidth availablility and free space in the routing table. If the node is too busy, drop the query and exit.
2. Determine if the desired resource is available locally; if so, enqueue the reply into the sender queue of the receiver that was specified by the query.
3. Decide how many nodes $n$ the query should be sent to (the query's time-to-live, information about the load, accounting information, and a random factor influence the decision); if $n > 0$, enqueue for sending to $n$ other nodes by doing the following:
   (a) Decide whether to replace the identifier of the previous requester with the local identifier based on current anonymity goals and load for the local node.
   (b) If the node replaces the identifier of the previous requester with its own identifier, associate the previous requester with the query in the "routing table".
   (c) Choose $n$ target nodes and enqueue the $n$ queries.

Peers also always:

- Flush individual queues (containing a mix of queries and replies) after a random (but bounded) amount of time.
- When receiving a reply, look in routing table for matching query and the identity of the next receiver. Enqueue the reply, or hand the content to the application layer for requests from local clients. Copy content into local storage (migration) if content is considered valuable and space is available.
- Discard infrequently accessed content from local storage.
- Send random content out into the network to provide background noise when network is idle (and to boost content migration)

### 2.3 Anonymity Guarantees

This section presents an analysis of how much anonymity peers can achieve using GAP. For the analysis, it is assumed that peers achieve perfect mixing of messages (queries and replies) in a given time interval. Perfect mixing in this sense means that an adversary cannot use timing analysis to correlate messages that the peer receives with messages that the peer sends. The time-interval for which this is achieved depends on the delay strategy that the peer is using, but optimal delay strategies for mixes are out of the scope of this work.

In order to answer the question of how strong the anonymity guarantees are that indirection can provide, some additional constraints must be considered. The first observation is that the more traffic a node creates, the more foreign traffic it must route in order to obscure its own actions. Suppose a node $A$ injects $n$ queries into the system and routes $m$ queries from other users. An adversary that does not participate in the network but monitors the encrypted traffic can see the amount $m$ of data that the node received and the amount $n+m$ of data that $A$ sent. Thus, this simple adversary would determine that any of the queries originated from $A$ with a probability of $\frac{n}{n+m}$.

If the adversary uses timing analysis, it is possible for the adversary to exclude certain amounts of traffic that were routed a long time ago. The interpretation of "long" here depends on the potential delay that a query may typically face in a node. Nodes can delay queries for random amounts of time in order to make this timing analysis harder. Excessively long delays make the query useless, and indirecting the query then becomes equivalent to producing noise (with the exception that other nodes will not perceive it as such).

Now, suppose that the adversary is actually actively participating in the network. In this case, traffic that originated from this active adversary cannot be included in $m$. The adversary knows that this traffic did not originate from the node under scrutiny. At this point, it should be clear why the assumption was made that every node always interacts with at least one node which does not collaborate with the adversary who is trying to break anonymity. Otherwise, $m$ could be zero and the probability of the node being identified as the originator would be $\frac{n}{n+m} = 1$ for any $n > 0$.

The degree of anonymity that can be achieved depends upon the power of the adversary and the amount of traffic that a node routes compared to the amount of traffic that a node generates. As long as the node is routing foreign traffic, the adversary can never be absolutely certain that this node is not the originator. The level of certainty at which the adversary will consider the identity of the originating node sufficiently established depends entirely on the specifics of the application at hand. The next section discusses how participants can individually trade anonymity for efficiency in order to make the degree of anonymity provided fit the needs of their individual applications.

### 2.4 Trading Anonymity for Efficiency

Suppose a node $A$ in GAP sends a query to node $B$. Now assume that $B$ forwards the query to a third node $C$. Furthermore, suppose that $B$ "maliciously"

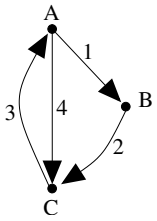



**Fig. 2.** Indirecting Replies

uses $A$'s address as the return address instead of its own. In this case, $C$ can determine that $B$ forwarded the query, and $C$ can eventually send the reply directly to $A$ (see figure 2). The term *indirect* is used if the node performs source identity rewriting. *forward* is used to indicate that the original sender identity is preserved.

Notice that while $A$ and $C$ now know each other, $C$ cannot be certain that the query originated from $A$, and $A$ cannot be certain that the reply originated from $C$. Both $A$ and $C$ could simply be nodes that obey the protocol and have indirected the query (or the reply). The anonymity of $A$ and $C$ depends upon how many packets they indirect for others; this amount of traffic is not changed by $B$'s "malicious" action. $B$ has not damaged the anonymity of $A$ or $C$. On the other hand, $B$ *has* potentially damaged its own anonymity. $C$ is now able to tell that this particular query did not really originate from $B$; all messages that originate from $B$ will have $B$ as the sender, as otherwise $B$ would never receive the reply. $C$ can discern that the message from $A$ is not in the set of messages that originate from $B$. By excluding this message from that set, $C$ can increase the probability that $B$ is the originator of any of the other messages that $B$ is currently sending. Thus, $C$ now has a higher chance of guessing which traffic actually *did* originate from $B$.

Since $C$ can also tell that $A$ is closer to the intended recipient of the reply than $B$, $C$ will send the reply directly to $A$. Because the reply takes the shorter path $C \rightarrow A$ instead of $C \rightarrow B \rightarrow A$, the total amount of traffic that was produced has been reduced. This "malicious" behavior of $B$ has improved the efficiency of the network. Note that $B$'s trade of anonymity for efficiency does not have any effect on $A$ or $C$, either in terms of anonymity or of efficiency.

While this technique improves bandwidth utilization and latency for the reply by saving one indirection, the performance gain could be even higher. Because $A$ and $C$ needed to communicate, $A$ may decide to send the next query directly to $C$. If $A$ is likely to send many related queries (related in the sense that the responses are likely to be located on the same node), it is reasonable to assume that $C$ will often be closer to the location of the document than $B$ is[1]. This way, the number of hops between $A$ and the content is decreased, speeding up the download process even further.

[1] The encoding of content in GNUnet [1] requires many related queries before a download of a single file can be completed. In other systems, queries may be related less often.

Let us suppose $B$ is indirecting $m$ queries and sending $n$ new queries for its own user. As stated above, this would yield a probability of $\frac{n}{m+n}$ that any given query originates from $B$. If $m$ is sufficiently large compared to $n$, this security may not be required by $B$. Indirecting $m$ queries and $m$ replies causes a great deal of work for $B$. If $B$ chooses not to indirect $k$ queries, and, instead forwards those queries preserving the original sender address, the probability that an adversary can assign to $B$ to be the originator of a query is increased to $\frac{n}{n+m-k}$.

## 3 Implementation

An implementation of GNUnet with GAP is available on our website at
`http://www.gnu.org/software/GNUnet/`.

### 3.1 Joining the Network

A node that wants to join the network must obtain a list of peers that is large enough to contain at least one non-adversarial node. The other node must be *non-adversarial* in the sense that it is not an attacker that will only advertise nodes that are entirely under an adversary's control (in that case, the adversary could keep track of what the new node is doing by making sure that it communicates exclusively with adversarial nodes). If the new node has several initial public keys of other nodes, it is sufficient if one of these does not collaborate with an adversary. For convenience, GNUnet can automatically download addresses of several network entry points from multiple http servers.

Each node in GNUnet has an RSA key pair. The nodes use these keys to exchange 128-bit session keys that are used to establish a link-encryption infrastructure between the nodes. Blowfish is used for the symmetric cipher. Nodes periodically sign their current Internet address (together with a time stamp for expiration) and propagate this information together with their public key. Except for the initial exchange of public keys that occurs when a node joins, this exchange of public keys can also use the encrypted channels.

### 3.2 Queries and Replies

Nodes indirect queries and can thereby hide the queries they originate since source rewriting makes all queries sent by the node look uniform. Every node uses a combination of network load and other factors that are internal to the node to determine how often to indirect queries. If the general network load is high, then the node indirects fewer queries, assuming that its own traffic is already well hidden. If the network load is low, more queries are indirected.

Several queries are usually sent out in a group, potentially mixed with other messages such as content replies or peer advertisements. Grouping several messages to form a larger packet introduces delays and decreases the per-message overhead. Encrypted packets containing queries are indistinguishable from packets containing other data because grouping and padding with noise makes them equivalent in size.

Each query contains the identity of the node where the reply is to be sent. While this was originally the address of the initiator of the query, nodes that indirect the query must change this sender identity to match their own. This is because these indirected packets could otherwise be distinguished (by the receiver) from packets that originate from the node itself, which have a different return address. The node must keep track of the queries that it has indirected so that it can send the reply to the node where the query originally came from. This statefulness of routing is probably the biggest scalability issue in GAP. Mix networks avoid this issue by keeping the state encrypted in the messages itself. The problem with this approach is that it requires slow public-key cryptography to encrypt the reply blocks. Also, query messages and replies have to accomodate encrypted reply blocks. The reply blocks would either have to be signed (increasing the number of public key operations and the message sizes even further) or would be subject to manipulations by malicious hosts, which could prevent nodes from properly routing the replies. Onion routing also addresses these issues with mixes by adding state (symmetric keys). Note that the number of anonymizing tunnels used in onion routing is typically smaller than the number of messages in GAP; thus, the problem with large routing tables is significantly smaller in the case of onion routing. For GAP, we decided that bandwidth and public key operations will presumably be the bottleneck rather than memory for routing information. The implementation of GAP contains various heuristics for estimating how long to keep entries in the routing table (based on time-to-live and importance of the query). Also note that when downloading a large file, not all queries for all blocks have to be parallelized.

### 3.3 Source Rewriting Is Optional in GAP

In GAP, the "malicious" behavior described in the section 2.4 is allowed. Nodes usually increase $k$ if they receive more traffic than they are willing or able to handle. Thus, if nodes receive a great deal of traffic, they can improve their performance by reducing the number of packets they indirect. Because the replies are significantly bigger than the queries, this behavior can improve the situation (particularly for bottlenecks in the network).

It is possible that this behavior could be exploited in an attack that uses flooding of a node, $A$, with traffic from a malicious node $M$ which tries to break $A$'s anonymity. As seen before, indirecting queries that originate from the attacker $M$ do not count toward $m$ in the formulas given above because the adversary knows that they do not come from $A$. If $A$ decides that the amount of traffic it gets is too high and then starts to preserve the sender addresses of most queries from other nodes, $m$ may decrease so far that $A$ can no longer protect its own $n$ queries from being discernible by $M$.

The same attack also applies to *mixes* [6] where adversaries that dominate the traffic at a mix can deduce the operation of the mix. GAP attempts to guard against this type of attack by dropping queries from nodes that are generating excessive amounts of traffic.

### 3.4 Choosing the Next Node

Whenever a GNUnet node receives a query, it decides how many nodes it will send the query to based upon its load (CPU, network), the local credit rating of the sender [10] and a random factor. The number of nodes that will be chosen to receive the query could be zero. The nodes that will receive the query are then chosen from the list of nodes that the node has established connections with (using a biased random selection process). The selection process is biased toward nodes where the hash of the hostkey is close to the query using some metric. This is a variant of the algorithm used by Pastry [15,2]. The selection process also takes into account recent network activity, with preference given to hot paths. Furthermore, queries are used to pad messages to uniform size, making use of bandwidth that would otherwise be wasted to transmit random noise.

The query is not sent immediately to the next group of nodes. Instead, it is put in a buffer that queues data that is to be sent to each selected node. The buffer is sent whenever it is full, or when a randomized timer goes off; it can also be entirely discarded if the node decides that it is too busy (note that the protocol does *not* guarantee reliable delivery).

This behavior does not directly leak any information to an attacker, as it is independent of the original sender; in fact, the originator has the same chance to receive the indirected query as everyone else has. Replies are sent back on the path that the query took originally, with the potential for shortcuts if intermediaries do not perform source rewriting and advertise another peer as the receiver of the reply.

Various attacks can be applied by a powerful adversary to almost any message routing scheme. One such attack would be a timing analysis that looks at the time that passes between query and reply. This time could be used to estimate the distance to the sender. GAP defends against this attack by introducing random amounts of delay for the query and the reply at every step. Furthermore, nodes choose the routes for a query at random, making the timing results hard to reproduce. Also, as with Freenet [5], content migration that can be caused by a request constantly changes the location of content, making it even harder to pinpoint its exact location.

Routing in distributed hash tables such as Chord [20] and Pastry [15] is subject to attacks where the adversary can predict the route of a query. The adversary can use the likelihood that a given peer will route a query to break anonymity. GAP makes this harder by adding significant amounts of variability to the routing process. One method GAP employs to add this variability is to modify the nature of cover noise sent out by peers. Instead of always generating and injecting noise into the network, peers also look at their lists of pending (unanswered) queries and choose queries to forward to additional hosts in the host list who have not yet been sent these queries by this peer. This suffices to make it plausible for any peer to receive and then route any query. The disadvantage is that GAP cannot guarantee $O(\log n)$ steps for finding a reply.

### 3.5 Looping Queries: Hops-to-Live and Time-to-Live

So far, one field that is contained in each query message, the time-to-live, has not been discussed. This field is used to bound the extent of the network that the query traverses. In particular, it is needed to prevent queries from looping indefinitely. More deterministic routing algorithms, such as Pastry [15] or Chord [20], do not have this problem since by design loops cannot occur. The more random nature of routing in GAP can cause queries to loop back to a peer that has already forwarded them. Thus, loop detection is a requirement to prevent queries from staying in the network for too long. Detecting loops by simply remembering which queries have been routed in the routing table is not feasible since the routing tables are typically unable to keep enough state for all queries in the network. Therefore, each query contains a field, the time-to-live, which bounds how long a query will be routed.

Freenet [5] uses a very similar scheme. Their routing algorithm deploys a hops-to-live field in every query. Every node on the path decrements the hops-to-live value by one until it reaches 1. Then, the query is forwarded only with a certain probability. This is needed to prevent an adversary from sending queries with the lowest hops-to-live value in order to see which nodes send replies; those that reply can be determined not to have forwarded the queries, since that would exceed the hops-to-live value, and the conclusion could then be drawn that nodes which reply are in fact the nodes storing the requested content. The Freenet scheme is problematic in that the adversary can use many probes to test whether a peer only replies with a certain probability or with certainty. While content migration is likely to actually always *make* the peer under investigation *a* node storing the content, this attack may still work if the adversary knows a set of correlated queries to probe multiple times with fresh content. In GNUnet, the block-encoding of the files would give any adversary an easy way to produce multiple independent probes to run this probabilistic attack.

For this reason, GAP does not use a hops-to-live field. The semantics of GAP's time-to-live field are slightly different. The time-to-live field is a relative time that specifies how long peers should route replies for this query. When a peer receives a query, it adds the time-to-live to its local time. This new absolute time is then used to produce a total ordering for all the queries that the peer receives. The fixed number of routing slots are assigned to the latest queries according to that total order. Note that the relative time-to-live field in a query can be negative, and that a peer may still route these queries if the replaced routing table entry is sufficiently old. The total order of the queries guarantees that a query can not loop in the network.

The implementation needs to be careful in routing replies to received requests; a node that is replying to a query $A$ should only send replies after the routing table slot for query $A$ has been allocated for long enough to make it plausible that this node has received a response for $A$ from another peer. Otherwise, the adversary could mount an attack where a series of queries $M_i$ is used to overwrite a routing table entry for $A$ (and its subsequent reply, of course). In this situation, the victim can only claim a plausible delay for the short time that $A$ was in the

routing table. Any additional delay can no longer deceive the adversary because responses to query $A$ from other peers could obviously not be routed back to the adversary after the routing table entry was replaced by one of the $M_i$s. GAP defends against this attack by sending replies only after the query has stayed in the routing table for an amount of time that makes it plausible for the peer to have received a reply from elsewhere. Thus, even if a node has the requested content, if $A$ disappears from the routing table, the content will not be sent. It should be noted that this solution may appear to leave the system open to denial of service attacks which would keep the node from being able to route any content at all. However, GNUnet's economic model [10] ensures that the traffic generated by flooding a node eventually causes the flooded node to drop traffic from the malicious node; thus, the only routing table entries which will be overwritten in such an attack will be those from other abusive nodes.

### 3.6 Measuring Anonymity

The discussion in this section assumes that the routing in GAP is sufficiently random to prevent the adversary from establishing that a node has a low probability of being chosen to route a specific query. Furthermore, the adversary is not supposed to be able to correlate queries (e.g. by the fact that they belong to the same file). This assumption is often reasonable since only by having the plaintext of the file it is possible to make that correlation. If the adversary does not know the exact details of the file that is being accessed, queries cannot be correlated.

From the explanation of how efficiency can be traded against anonymity in section 2.4, it should be clear that the degree of anonymity that GAP offers is configurable. If a node injects a query from the user into the network in only 1 of 1000 indirected transactions, it will surely be more anonymous than if it does so in 1 out of 10. Note that there are two parameters here. First, a node determines how much bandwidth it has available. Once that bandwidth quota is exceeded, is ceases to perform source rewriting on the additional traffic. Next, it chooses how anonymous a specific download or search needs to be. Given the currently available traffic (which may be anywhere between the minimal amount of background noise and the available bandwidth), it can then inject the queries at a certain frequency into the network.

Figure 3 illustrates this relationship. Suppose that the noise in the network that is routed through the node is at least $L$ kbps and that there are no active adversaries contributing to the traffic in the network. If a node on the network sends requests at a constant rate $r$, the probability $p$ that a packet originates from the node is $\frac{r}{L} = p = \frac{1}{X}$. If the available foreign traffic on the network increases above the basic noise level, the anonymity of the node also increases. At some point, the network traffic may reach the capacity $M$ of the node; the node stops routing the additional traffic, and the maximum degree of anonymity $Y$ that the node can achieve (while sustaining a data rate of $r$) is $\frac{r}{M} = \frac{1}{Y}$. If a node needs more anonymity than the current volume on the network can guarantee, it must slow down the rate $r$ at which it requests data from the network.

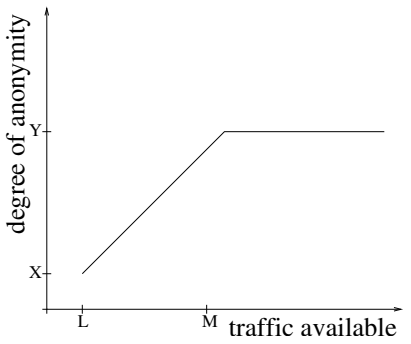



**Fig. 3.** Anonymity and Traffic

With adversaries that have unknown power, the exact degree of anonymity and the probability with which the adversary can determine the originator cannot be computed in this manner in practice. The reason is that this would require knowledge about how much traffic is controlled by the adversary. In fact, the more bandwidth an adversary has, the less anonymity can be provided since the adversary can flood the network with messages. This attack also applies to other anonymizing networks [6]. As stated in section 1, if the adversary controls the traffic of the entire network, it can always determine with certainty from where an action originated.

Estimating the number of adversary-controlled hosts in an open network like GNUnet is obviously very difficult. Proving the authenticity of a remote microprocessor [11] and (ultimately) the trustworthiness of a remote machine are still open problems. GNUnet compensates for the impossibility of guaranteeing anonymity against very powerful attackers by providing deniability [1]. Even if a powerful adversary can determine who sent a message, the deniable encoding and searching mechanism for content (see [1]) ensures that the adversary may still be unable to determine what the message is about.

A situation similar to that in which an adversary floods the network with known traffic occurs when there is only the minimal amount of traffic $L$ on the network, allowing for only a minimal degree $X$ of anonymity. This situation is not as problematic as it may sound since peers can start with a very slow download rate $r$. This will increase the network load on the network, especially since idle nodes are likely to spread a query much further than busy nodes would. Thus the load on the network will quickly rise (at least in the proximity of the peer starting the download), allowing the peer to increase $r$. Since GNUnet's content encoding [1] has the inherent property that a downloading node can initially only send a small set of queries (due to the tree-structure of the encoding), the requirement that a node must start with a small $r$ to achieve anonymity until the network load rises is in practice what the code must do anyway. The network always has some level of background noise for key exchange and node advertisement propagation that should be sufficient for a node to hide the origin of a single query.

## 4 Related Work

Indirection, delays and noise have been used to achieve anonymity long before GAP [3,21,12]. Interestingly, the traditional perception has focused on decoupling the identity of the receiver from the identity of the responder by placing an anonymizing service like `anonymizer.com` in the middle. While this approach works for weak adversaries like web-site operators that only see the intermediary's IP in their logs, it does not help against adversaries that can perform traffic analysis and even become a part of the anonymizing infrastructure.

Indirecting *all* communications is also very costly. For example, in Freenet [5], the number of indirections is determined by the length of the search path that the query takes until a node that has the content is found. Thus, if the search path has length $l$, there are $l$ transfers of the content. The traffic overhead is then approximately $(l-1) \cdot s$ where $s$ is the size of the content. Freenet attempts to counter this problem by using clever routing and content migration strategies to minimize $l$. The design does not allow the user to trade anonymity for efficiency.

Other systems, like Crowds [13], allow the user to set the number $l$ of indirections that the system should aim for. While the traffic overhead is again $(l-1) \cdot s$, the $l$ can be adjusted. The authors describe a network where $n$ nodes indirect requests with a probability $p_f$. The degree of anonymity that Crowds offers is defined as the probability that a node collaborating with the adversary receives a communication from the node that actually sent the request. As with GAP, the (unknown) strength of the adversary makes it impossible in practice to determine the exact degree of anonymity that is achieved.

The analysis of Crowds assumes that all nodes are equally active and thus equally suspicious. Even if the adversary has only $c$ nodes under control, traffic analysis may give much better data about which node is responsible for the query – even under the assumption that traffic between the $n-c$ non-malicious nodes cannot be decrypted. Sending noise to make the traffic analysis harder is not discussed and would, of course, increase the network load beyond $(l-1) \cdot s$.

In Crowds [13], a probabilistic attack is described that can be used to infer the identity of the initiator. The attack is based upon the idea that an adversary could attempt to correlate multiple transfers over the network and then infer the initiator who would have a higher-than-average chance of being the sender. Hordes [19] attempts to make this attack more difficult by using multicasts for replies, choosing a different path through the network from initiator to responder than the path back from responder to initiator. The probabilistic attack described requires not only that the adversary control nodes that are participating in the network, but also that the adversary is able to relate multiple transactions. In GNUnet, transactions cannot be correlated since neither the query nor the reply reveal any context unless the adversary knows exactly what content is transmitted. Thus, in this case, a probabilistic intersection attack over time will not help to reveal the identity of the user.

Another distributed network with anonymity goals is P5 [18]. P5 uses broadcasts (or multicasts) for data transfer. It achieves scalability by dividing the network into a tree-hierarchy of small broadcast groups. For sender-anonymity,

P5 nodes constantly broadcast noise in order to disguise active phases. Receiver addresses in P5 are addresses of these broadcast groups. A peer in one of the broadcast groups can advertise any group that is located higher up in the tree as its address. Each group forwards all received messages to all child-groups. Messages are dropped if the network load gets to high. Messages that have been addressed to a less specific group will be dropped earlier. Thus by advertising a less specify broadcast group the anonymity of a peer can be increased since the number of groups that may receive the reply is larger. Advertising a more specific group on the other hand improves the latency of the peer since fewer messages will be dropped. The overall traffic in P5 is assumed to be always at the maximum of what the peers can sustain. Thus P5 allows peers to trade-off anonymity for latency but not for bandwidth.

DC-net is an anonymous network based on the Dining Cryptographers protocol [4] that requires each node to communicate with each other node for every message. Onion Routing is based upon Chaum's mixes [3]. For each message, the user chooses a path though the mix-network and encrypts the message with the public keys of each of the mixes. The message is then sent along that path and as long as not all nodes on the path are compromised, the identity of the initiator is not directly exposed.

In most of the networks above, the anonymity of a node depends upon the behavior of the other nodes in the network. In GAP, each node is able to individually choose whether to exchange portions of its own anonymity for its own efficiency without impacting the security of other nodes.

### Comparing Anonymous Protocols

We now compare GAP to related anonymity protocols, focusing on P5 [18], Freenet [5], DC-Net [4], mixes [3], Onion-Routing [21], Crowds [13], Hordes [19] and a simple proxy. Note that the designs compared here address very different applications, from interactive anonymous browsing to high-latency anonymous mail. The systems differ widely in their respective costs and benefits and it is thus difficult if not impossible to make a fair comparison.

P5, DC-Net and Hordes rely on broadcasts or multicasts for communications; all of the other protocols use unicast. GAP and Freenet use unicast, but nodes may choose to route the same message to multiple nodes (which could be seen as application-level multicast); however, these duplicates are actually *processed* by each recipient (as opposed to most anonymizing multicast schemes, in which every recipient but one just sees the traffic as noise).

Most anonymous protocols achieve at least some form of initiator (or sender) anonymity. In the case of a proxy, the sender is only anonymous in relation to the responder, not in relation to the proxy, as the sender's identity and data are directly exposed to the proxy. In mix networks, if the initiator is not part of the mix network, initiator anonymity is again only partial for the same reason. In these anonymizing networks, there is a single point of failure (typically the first hop) at which a node controlled by the adversary can be used to fully expose the initiator. One could argue that the message is encrypted and can only

be decrypted by the final recipient and, thus, that anonymity is not violated. But even when the contents are not exposed, it is still possible to correlate the transaction with the sender. While our definition of anonymity allows that the adversary may see participation in the network, it does not allow the adversary to determine that a peer actually initiated a transaction.

In P5, Freenet and GAP responder anonymity is achieved in addition to sender anonymity. This is because the reply gets anonymized as it goes back through the network and the responder is not known to the initiator. The responder is also anonymous in DC-Nets, since the protocol only deals with one-way broadcast messages (symmetry). Mixes and Onion Routing can have anonymous responders when reply-blocks are published, but not all responders are anonymous by default. Crowds and Hordes require prior knowledge of the responder by the initiator, thus making responder anonymity impossible unless a public rendezvous point is used [16]. Mixminion [6] discusses attacks on mix networks.

Some protocols allow nodes to trade anonymity for efficiency in order to improve performance. With a simple proxy, DC-Net or Freenet, the degree of anonymity is fixed by the circumstances. In Crowds and Hordes, a node can choose to reduce the number of indirections on the communications path (or the size of the multicast group in the case of replies in Hordes), increasing the efficiency of the network. Note that a node that reduces its number of indirections reduces the load on other nodes in the network, but not the load on itself (thus reducing incentive to make this trade-off). Also, joining a larger multicast group in P5 or Hordes affects other nodes that will receive additional useless multicast traffic. With GAP, a node that indirects a query but tells the recipient to shortcut the response actually is able to reduce its own load (since it does not have to route the reply) without having an impact on the anonymity or load of any other node.

Attackers that actively participate in the protocol are hard to defend against. The best defense is a DC-Net where a single non-collaborating node can thwart an attack against any node except itself. In the case of a proxy, the only node in the network must be trusted; thus, a "collaborative" attack will always succeed. In P5, an adversary must control the respective multicast group (which can be small) in order to ascertain the identity of the recipient. Freenet, Onion Routing and Crowds (and to a lesser degree Hordes) are vulnerable to the probabilistic attack over time described in [13,19]. In GAP, the adversary must control a significant portion of the bandwidth that a node is routing in order to be able to determine with reasonable probability which traffic was initiated by that node.

Freenet and GAP are anonymity protocols that have built-in content location capabilities. P5 requires knowledge about the address of a multicast group containing the responder. All other systems compared require prior knowledge by the initiator about the address of the responder.

P5 and Mixes require one or more public key operations *per request* on each node processing the message; the other systems require only symmetric operations for each request after an *initial* public key exchange that is used to establish link encrypted channels.

### 4.1 Other Anonymizing Systems

Tarzan [9] is a peer-to-peer system that allows IP-level anonymization of Internet traffic based on onion routing. Tarzan cannot offer responder anonymity. Worse for the user is probably the fact that applications that use Tarzan are typically not aware of the anonymization requirements. Users are likely to use Tarzan in combination with applications such as web-browsers or mail clients that will often allow the responder to identify the user due to information leaked in the higher-level protocol that is tunneled in the anonymizing IP layer infrastructure.

Morphmix [14] improves on mixes by using a witness to help select the path through the network. In this way, the initiator does not need to know a large set of peers in order to build an anonymous tunnel to the receiver. Like Tarzan, Morphmix is a peer-to-peer network where the set of mixes is large and dynamic, as opposed to static sets that were used in previous architectures.

Peer-to-peer networks like Tarzan, Morphmix and GNUnet are faced with the problem that there is no operator that is responsible for keeping the network running. This makes these open networks an easy target for adversaries that are powerful enough to disrupt Internet connections. Individual peers maybe isolated from all other peers on the IP level and be only allowed to connect to nodes controlled by the adversary. Peer-to-peer users would probably not even notice this type of attack since peers are always considered to be unreliable, thus not being able to reach a large fraction of the peers would be considered normal. GNUnet attempts to make this attack harder by providing a transport abstraction that can tunnel GAP messages in other protocols, such as SMTP [8], making it harder for the adversary to disrupt all communications without being noticed.

### 4.2 Measuring Anonymity

GAP's model for anonymity is probability based. [7,17] have described why an approach based on probabilities does not prevent individual peers from sticking out. A peer that has only a probability of 5% to be the originator of a query may still be an easy target if all other peers have a significantly lower probability, say 1%. [7,17] proposed an entropy based anonymity metric that takes the probability distribution into account. In this information theoretic approach, the resulting metric expresses how many bits of information the adversary would need to expose a peer. It turns out to be difficult to apply this metric to GAP since computing the entropy of the network requires a global view of the network that typical peers do not have. Thus peers can only attempt to reduce their individual probabilities, but they can not do this based on knowledge about the global distribution.

The entropy based metric shows that all anonymizing protocols need to attempt to balance probabilities as much as possible. While this requirement was taken into account when GAP was designed, future work will be needed to formally determine the amount of information leaked by the protocol for a given type of adversary.

## 5 Future Work

The discussion of GAP in this paper has focused on a single query sent by the initiator. In practice, downloading a file in GNUnet will require multiple queries. While an active adversary as described in the introduction cannot correlate these queries, a content-guessing global active adversary that has obtained the file that the user is downloading (e.g. by downloading it for himself) will be able to correlate these queries. If the adversary is able to correlate queries, it may be possible for the active adversary to infer the identity of the initiator probabilistically. Better defense against this type of attack is an open issue.

In this paper we have also assumed that the mixing performed by the peer is *optimal*. While improving mix algorithms is basically an orthogonal issue to indirection-based anonymization protocols, the routing decisions made at each peer impact the mixing strategy; this makes this type of mixing slightly different in comparison to traditional mix networks in which the next hop is determined entirely by the message itself. In general, scalable anonymous routing strategies are still an open problem. We hope to evaluate the scalability of the routing strategy presented in this paper in the future.

Content migration is another issue. If a peer sends out local content that does not correspond to any query that it has recently received, the recipient knows that the sender stored the content. One solution to this is to randomly forward content on a frequent basis that the peer received but did not store locally. Another approach may be to forward content only when the peer is about to remove it from its local store. The use of onion routing for content migration may be another solution which is more costly, but less limiting.

## 6 Conclusion

This paper presented an adaptive indirection scheme that allows each node in an anonymizing network to individually trade anonymity for efficiency without negatively impacting other nodes in the network. A new perspective on how to perceive anonymity from the perspective of strong adversaries that monitor all network traffic and potentially even participate in the anonymizing network was described. This perspective was then used to derive an anonymizing protocol which allows participants to choose not to perform source-rewriting (in order to increase efficiency) without violating the protocol. The implementation of GAP in GNUnet uses the current network load to exchange increasing anonymity for efficiency, allowing the improvement of scalability.

A final thought: while trading anonymity for efficiency may mean being less anonymous in the short term, increased network efficiency may mean greater usability. This may lead to more users and therefore eventually to *increased* anonymity.

## Acknowledgements

We would like to thank the anonymous reviewers for constructive feedback and Roger Dingledine for helpful discussions. Additionally, we are grateful to Jan Vitek and Victor Raskin for support.

## References

1. Krista Bennett, Christian Grothoff, Tzvetan Horozov, and Ioana Patrascu. Efficient Sharing of Encrypted Data. In *Proceedings of ACISP 2002*, 2002.
2. M. Castro, P. Druschel, Y. Hu, and A. Rowstron. Exploiting network proximity in distributed hash tables. In *Proceedings of the International Workshop on Future Directions in Distributed Computing*, 2002.
3. David Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.
4. David Chaum. The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. *Journal of Cryptography*, pages 65–75, 1988.
5. Ian Clarke, Oscar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. *Lecture Notes in Computer Science*, 2009, 2000.
6. George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *IEEE Symposium on Security and Privacy*, 2003.
7. Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In *Proceedings of the Workshop on Privacy Enhancing Technolgoies*, 2002.
8. Ronaldo A. Ferreira, Christian Grothoff, and Paul Ruth. A transport layer abstraction for peer-to-peer networks. In *Proceedings of GP2PC 2003*. IEEE Computer Society, 2003.
9. Michael J. Freedman and Robert Morris. Tarzan: A Peer-to-Peer Anonymizing Network Layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, Washington, D.C.
10. Christian Grothoff. An Excess-Based Economic Model for Resource Allocation in Peer-to-Peer Networks. *Wirtschaftsinformatik*, June 2003.
11. Rick Kennell. Proving the Authenticity of a Remote Microprocessor, 2003.
12. Michael Reed, Paul Syverson, and David Goldschlag. Proxies for anonymous routing. In *12th Annual Computer Security Applications Conference*, pages 95–104, December 1995.
13. Michael K. Reiter and Aviel D. Rubin. Crowds: anonymity for Web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.
14. M. Rennhard and B. Plattner. Introducing MorphMix: Peer-to-Peer based Anonymous Internet Usage with Collusion Detection. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES), in association with 9th ACM Conference on Computer and Communications Security (CCS 2002), Washington, DC, USA*, November 2002.
15. Antony Rowstron and Peter Druschel. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. *Lecture Notes in Computer Science*, 2218, 2001.

16. Vincent Scarlata, Brian Levine, and Clay Shields. Responder anonymity and anonymous peer-to-peer file sharing. In *Proceedings of IEEE International Conference on Network Protocols (ICNP) 2001.*, 2001.
17. Andrei Serjantov and George Danezis. Towards an information theoretic metric for anonymity. In *Proceedings of the Workshop on Privacy Enhancing Technolgoies*, 2002.
18. Rob Sherwood, Bobby Bhattacharjee, and Aravind Srinivasan. P5: A Protocol for Scalable Anonymous Communication. In *IEEE Symposium on Security and Privacy*, 2002.
19. Clay Shields and Brian Neil Levine. A protocol for anonymous communication over the Internet. In *ACM Conference on Computer and Communications Security*, pages 33–42, 2000.
20. Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications*, pages 149–160. ACM Press, 2001.
21. Paul Syverson, David Goldschlag, and Michael Reed. Anonymous Connections and Onion Routing. In *IEEE Symposium on Security and Privacy*, pages 44–54, Oakland, California, 4–7 1997.

# An Analysis of GNUnet and the Implications for Anonymous, Censorship-Resistant Networks

Dennis Kügler

Federal Office for Information Security
Godesberger Allee 185-189
53133 Bonn, Germany
Dennis.Kuegler@bsi.bund.de

**Abstract.** Peer-to-peer networks are a popular platform for file sharing, but only few of them offer strong anonymity to their users. GNUnet is a new peer-to-peer network that claims to provide *practical* anonymous and censorship-resistant file sharing. In this paper we show that GNUnet's performance-enhancing features can be exploited to determine the initiator of a download. We also present an efficient filter mechanism for GNUnet. Assuming that content filtering is legally enforced, GNUnet can be censored at a large scale.

## 1 Introduction

Peer-to-peer networks are widely used to share all kinds of digital content with other users. It was first demonstrated by Gnutella [FP00] that such peer-to-peer networks can be organized in a decentralized manner, so that there is no practical way to shut down the network. Therefore, Gnutella and related networks have become a popular platform for sharing legal and illegal content (i.e. copyrighted or subject to censorship). While searching for content on Gnutella is relatively anonymous, responses to search queries are non-anonymous as the IP address of the offerer is always exposed. Thus, it is possible to prosecute users who break the law. As this clearly discourages users to share illegal content, censorship is indirectly applied.

The idea of constructing a censorship-resistant network goes back to the idea of the Eternity Service [And96], an attack-resistant storage medium. Freenet [CSWH01,CMH+02] was the first approach that tries to combine both types of networks: an anonymous, decentralized peer-to-peer network and a censorship-resistant network. Freenet can perhaps be described best as "distributed file system storing replicated content in an obfuscated form". One obvious drawback of Freenet is that it "forgets" infrequently requested content, which not only contradicts censorship-resistance but also makes the retrieval of content inconvenient as downloads often fail.

Recently, GNUnet [GPB+02,BGHP02,BG03,Gro03] has been proposed, an alternative approach that claims to be much more practical and efficient. Therefore, we analyze GNUnet in this paper and point out some weaknesses. We start

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 161–176, 2003.

in Section 2 with a short introduction to GNUnet. Then we show how anonymity can be degraded: We introduce our shortcut attack in Section 3 and consider the efficiency of this attack in Section 4. Afterwards, we discuss in Section 5 how censorship can be applied to GNUnet. Finally, we conclude our paper in Section 6.

## 2 A Description of GNUnet

GNUnet consists of nodes that communicate with each other. Every node chooses a key pair, that is used for identification, authentication, and to encrypt the communication between the nodes. To hide which nodes are indeed communicating with each other, GNUnet uses a MIX-like [Cha81] approach: nodes are intermediaries that send messages to other nodes.

In the following, we present the encoding scheme and the routing mechanism of GNUnet, which will both be exploited for our attacks.

### 2.1 Encoding[1]

GNUnet transfers blocks of fixed size (1Kb). There are three types of blocks: *Data Blocks* (DBlocks), *Indirection Blocks* (IBlocks), and *Root Blocks* (RBlocks).

Files of arbitrary size are split into DBlocks. For every DBlock $D_i$ of a file the 160 Bit RIPE-MD hash value $H(D_i)$ is calculated. Then a tree of IBlocks is calculated recursively, where every (leaf) IBlock contains up to 25 query-hashes (see below), a superhash and a CRC32 checksum. The superhash of an IBlock is calculated as the hash of the concatenation of all query-hashes in the blocks below this IBlock. Every block $B_i$ is encrypted with the hash value of its own content to $E_{H(B_i)}(B_i)$ and is stored under $H(E_{H(B_i)}(B_i))$. Thus, the query-hashes included in the IBlocks are pairs of the form $(H(B_i), H(E_{H(B_i)}(B_i)))$ that are necessary to retrieve the remaining blocks.

Finally, the RBlock is generated, containing a description of the file and the query-hash to retrieve the root of the tree of IBlocks. One or more keywords are used to encrypt and to store the RBlock. For every keyword $K_j$ the RBlock $R$ is encrypted to $E_{H(K_j)}(R)$, then both the encrypted RBlock and $H(H(K_j))$ are stored under $H(H(H(K_j)))$.

### 2.2 Routing

Retrieving content from GNUnet is a two step process:

1. **Discover RBlocks:** RBlocks are discovered using search queries containing a list of triple hashed keywords $H(H(H(K_j)))$. A node that receives a search query and has a RBlock stored under one of those values returns the encrypted RBlock $E_{H(K_j)}(R)$ together with $H(H(K_j))$ to prove that it indeed possesses the requested content stored under this keyword. The node that has issued the query is able to decrypt the RBlock using $H(K_j)$.

[1] The encoding scheme used by GNUnet to store and retrieve files has changed recently. The original scheme can be found in [BGHP02].

2. **Download Content:** After an RBlock is discovered, the node can download the corresponding file by issuing several download queries. The node first uses the query hash included in the RBlock to retrieve and decrypt the root IBlock. Then the node uses the query-hashes included in the root IBlock to retrieve and decrypt all additional IBlocks and finally, the query hashes included in the leaf IBlocks are used to retrieve and decrypt the DBlocks of the file. The node may also use multi-queries to retrieve several I- or DBlocks at once. A multi-query consists of several download hashes plus the corresponding superhash to speed up lookups.

Thus, there are two types of queries: search queries and download queries. Both types of queries contain three additional values that are used to process the query: a return address, a priority, and a time-to-live (TTL).

**Processing Queries.** Each node sets up a message queue for every neighbor node. Before a query or a response is sent to a neighbor node, it is temporarily stored in the queue representing this node. Each queue is flushed at random intervals, then all messages waiting in this queue are sent to the corresponding node. A node that receives a query proceeds as follows:

1. If the requested content (RBlock, IBlock or DBlock) is found locally, a response is enqueued in the message queue corresponding to the return address given in the query. Otherwise (but always in the case of a search query), the query is sent to some other nodes.
2. Depending on the priority and the available bandwidth the node randomly selects $n$ neighbors and enqueues the query in their message queues. The query is adjusted as follows:
   - The new priority is calculated as $\frac{p}{n+1}$, where $p$ is the old priority.
   - The node either keeps the return address of the predecessor node or replaces it with its own address. In the latter case, the node adds the query to its local routing table.

It is important to mention that intermediary nodes are not aware which queries belong together (with the exception of multiqueries). Therefore, queries are routed individually, i.e. there is no static path that is used to route related queries. However, the selection process is biased towards neighbors that have responded recently. This is discussed in more detail in Section 4.

**Credit.** GNUnet is based on an economic system, where every node associates any other known node with a local value called *credit*. The goal of this economic system is to prevent flooding attacks by limiting the resources available to an attacker:

Let $A$ and $B$ be two nodes. $A$ associates $B$ with credit $c_B$ and vice versa. After $B$ received a query with priority $p$ from $A$, it first checks whether $A$ has enough credit.

- If $c_A = 0$ the query is dropped, otherwise set the new priority $p'$ to $p$.
- If $p' > c_A$ the priority of the query is reduced to $p' = c_A$.

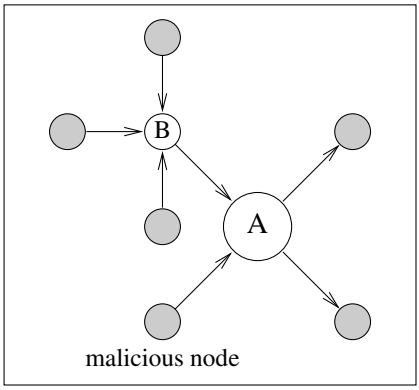



**Fig. 1.** Malicious nodes in a network.

Then $B$ processes the query and charges $A$ by reducing $A$'s credit to $c_A = c_A - p'$. However, if $B$ has excess bandwidth, it may decide not to charge $A$. The decision whether and how much $B$ decreases $A$'s credit is invisible to $A$.

If $B$ returns a valid response, $B$ expects $A$ to increase his credit to $c_B = c_B + p$. Again, $A$ may decide not to pay $B$ for returning a valid response. The decision whether and how much $A$ increases $B$'s credit is invisible to $B$.

**Zero Priority Queries.** If a node indirects a query to another node without charging the preceding node, it may assign zero priority to the forwarded query, because it does not want to be charged by the following node.

Thus, a node that receives a zero priority query cannot gain any credit when responding to this query. But if the load on the node is low, responding to a zero priority query does not hurt this node.

## 3 Deanonymization in GNUnet

An informal definition of anonymity can be found in [PK01]:

> *"Anonymity is the state of being not identifiable within a set of subjects, the anonymity set".*

We consider a network of several nodes, where some nodes may be malicious as shown in Figure 1. A malicious node is behaving correctly, but it tries to acquire as much information about the network as possible. Furthermore, we assume that all malicious nodes are controlled by the same attacker.

### 3.1 Identifying Sessions

To successfully apply our attacks, we first have to identify sessions. This can be done very easily: GNUnet splits larger files into small DBlocks. If we know the corresponding IBlocks, we are able to identify DBlocks that belong together.

Thus, the attacker prepares a dictionary of interesting keywords and their corresponding hash values, queries the network with those keywords and receives a number of encrypted RBlocks $E_{H(K)}(R)$ which he is able to decrypt. Then the attacker can also retrieve the corresponding IBlocks. As the IBlocks contain the query-hashes of all DBlocks, the attacker is able to use his malicious nodes to observe the following:

- Queries containing one of the prepared keywords.
- Queries for known I- or DBlocks.
- Responses containing known I- or DBlocks.

**Note:** The attacker can also observe responses to queries with non-trivial keywords if the file also has been inserted (probably by a different user) under trivial keywords. Inserting content under a non-trivial keyword renders the content useless as only few people are able to retrieve the content again. This is a contradiction to the idea of a censorship-resistant network, where everybody should be able to access any possible content. Therefore, we expect that most content is available to the attacker.

An attacker can exploit the linkability of related I- and DBlocks to execute two types of attacks on the anonymity:

1. **Intersection Attacks:** The intersection attack [BPS01] exploits the fact that not all users of a MIX network participate in every batch. Thus, all users that have not contributed to a batch containing linkable traffic, can be subsequently removed from the anonymity set.
2. **Predecessor Attacks:** The predecessor attack [WALS02] extracts information from the setup of dynamically chosen, linkable paths, where each node randomly selects the following node out of the set of all nodes. To determine the initiator, the attacker logs the preceding node. Besides the initiator, every node should be logged with equal probability. Thus, the expected number of times the initiator is logged is greater than the expected number of times any other node is logged.

The predecessor attack has been used in [WALS02,Shm02] to successfully attack Crowds [RR98]. In GNUnet however, the assumption that every node is logged with equal probability is not satisfied, as each node only sends queries to its direct neighbors, not to all nodes. Therefore, we will not discuss this attack. However, if the attacker should be able to establish a connection to all nodes in the network, then he will receive the same query from many nodes. In this case, the predecessor attack should be reconsidered, as the attacker will receive the query more often from the initiator node than from any other node.

### 3.2 The Shortcut Attack

Our *shortcut attack* is a special intersection attack that exploits GNUnet's shortcut feature. Depending on the current load, a node uses this feature to decrease both bandwidth utilization and latency:

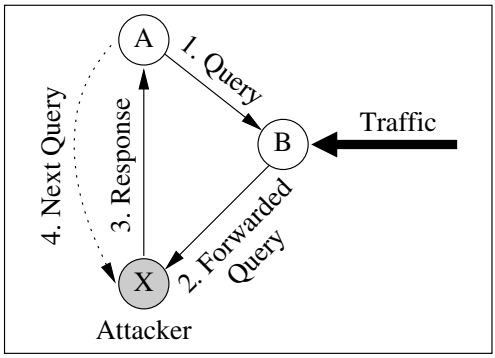



**Fig. 2.** The shortcut attack.

- If the node is idle, queries are *indirected*. The node replaces the return address with its own address.
- If the node is busy, queries are *forwarded*. The node does not change the return address of the preceding node.
- If the node is very busy, queries are *dropped*. The node does not process the query.

Each node monitors its own outbound traffic to determine how busy it is. A node that has too much outbound traffic simply optimizes the routing of responses by removing itself from the path.

It is claimed by the authors of GNUnet that shortcuts do not hurt anonymity: A node that stops indirecting some traffic can only hurt itself, as hiding its own activities is worse with lower (outbound) traffic. We exploit this optimization to discover the initiator of a query as shown in Figure 2, where node $X$ is an attacker who receives queries from node $B$ that have been issued by node $A$.

**Basic Shortcut Attack.** The attacker tries to entice an attacked node to forward traffic instead of indirecting it by increasing the node's outbound traffic:

1. The attacker floods $B$ with high priority queries. Due to the promised credit $B$ will try to respond to those queries. The attacker may hide his attack by using several malicious nodes to flood $B$.
2. The load of $B$ increases and the node decides not to indirect but to forward (or to drop) some queries. When $B$ forwards queries, it does not overwrite $A$'s return address, so that the attacker will learn that $A$ is the node preceding $B$.
3. The attacker directly responds to $A$. As the attacker is providing valid content, $A$ will send further (perhaps unrelated) queries to the attacker. The more the attacker responds to $A$'s queries, the higher is the probability that $A$ sends more queries to the attacker.

The basic shortcut attack is very inefficient as the attacker is not able to control which query the attacked node forwards. Given that the attacked node may be

connected to hundreds of other nodes, it is very unlikely that just the query the attacker is interested in is forwarded to him. Furthermore, credit is required at the attacked node to increase its load. As credit is not global, the attacker has to gain credit at each attacked node before the attacker can proceed. To gain credit at a node, the attacker has to respond to several queries.

Thus, the basic shortcut attack is only applicable, if paths are very static. As this is not the case, we have to remove the requirement to flood the attacked node.

**Improved Shortcut Attack.** A precondition for the improved shortcut attack is that the attacker is somehow aware of the topology of the network. Then the attacker can simply guess that $A$ is closer to the initiator than $B$. The attacker therefore connects to $A$ and returns responses to queries received from $B$ directly to $A$. If the attacker has guessed correctly, then he provides valid content to $A$ and attracts more queries from this node. Furthermore, $B$ is perhaps unable to respond to $A$'s queries without the help of the attacker, which again will increase the probability that $A$ indirects more queries to the attacker.

## 4 Applying the Shortcut Attack

To apply the improved shortcut attack, the attacker uses the following strategy get subsequently closer to the initiator:

1. The attacker connects to all neighbors of the current preceding node and waits for linkable queries.
2. The attacker uses a statistical test to determine the neighbor that is closer to the initiator.
   (a) If a closer node exists, it is selected as new current node.
   (b) Otherwise the current node must be the initiator.

In the following, we show how the attacker can decide which of the tested nodes is closer to the initiator. As the success of the test depends on how GNUnet routes queries, we first discuss the efficiency of the attack when only random routing is used, which is GNUnet's basic routing mechanism. Then we show that the attack becomes much easier when GNUnet tries to improve performance and additionally uses hot path routing.

### 4.1 Testing Neighbor Nodes

Assuming that the attacker receives linkable queries with a higher probability from the node that is closer to the initiator than from the other tested nodes, the attacker can use a distribution-independent statistical test (e.g. Wilcoxon-Mann-Whitney or Run-Test) based on the sequence of received linkable queries to determine the node that is closer to the initiator.

**Exact Test.** For each pair of nodes $(A, B)$ we have to decide, whether both nodes are equal likely to receive linkable queries (null hypothesis) or not (counter hypothesis). Thus, we have the following hypotheses:

$$H_0\colon P_A = P_B$$
$$H_1\colon P_A \neq P_B$$

To be able to accept or reject $H_0$ with significance $\alpha$, several linkable queries have first to be received from those nodes. If the attacker rejects $H_0$ he is only able to state that both nodes are not equal likely to receive linkable queries. But as we assume that the closer node receives more linkable queries, the attacker chooses to continue the attack with the node that has delivered more linkable queries.

**Simplified Test.** Although the exact test should provide good results even if it is difficult to distinguish the tested nodes ($P_A \approx P_B$), the attacker may want to use a simplified test. The exact test is expensive for the attacker as several linkable queries first have to be received from the tested nodes before the attacker can accept or reject $H_0$ with significance $\alpha$.

If we additionally assume that $P_A \gg P_B$, if $A$ is closer to the initiator than $B$, we can simplify the test. The attacker guesses that the node that first delivers a linkable query is closer to the initiator. The attacker chooses this node to continue with, but has to verify his guess afterwards. Therefore, the attacker has to monitor the tested nodes until enough queries are received to either accept or reject $H_0$ with significance $\alpha$. If it turns out that the attacker has to unexpectedly accept $H_0$, the attacker has to chose another node based on the results of the exact test, because his guess was wrong.

### 4.2 Random Routing

With random routing the nodes to receive a query are selected randomly. Let $k$ be the number of neighbors of node $A$. For every query the node randomly chooses $m$ of the $k$ neighbors and sends the query to them. Thus, the probability that a certain neighbor is randomly selected is uniformly $P_{rnd}(A) = m/k$ for all neighbors of $A$. Note that the probability $P_{rnd}$ is node-specific and unknown to the attacker.

**Testing.** To test which of the neighbor nodes $B_i$ of the current preceding node $A$ is closer to the initiator, we compare the number of queries received from those nodes. Let $q_A$ be the number of linkable queries the attacker receives from $A$. Then the attacker can expect to receive the following number of linkable queries from a tested node $B$ in the same time interval. If the tested node $B$ is closer to the initiator, the attacker expects to receive

$$q_B = q_A / P_{rnd}(A)$$

linkable queries from this node. Otherwise, the tested node is more distant from the initiator and the attacker expects to receive only

$$q'_B = q_A \cdot P_{rnd}(B)$$

linkable queries from this node.

It is important that $P_{rnd}(A)^{-1} \gg P_{rnd}(B)$ to succeed with the simplified test. Therefore, the smaller $P_{rnd}$ is, the better can the attacker use this test to distinguish which node is closer to the initiator.

- In general $P_{rnd}$ is very small in GNUnet, as every node has many neighbor nodes but only few of them are randomly selected. In this case the node that is closer to the initiator can be determined efficiently.
- However, if the load of a node is low, the node begins to pad the empty message queues with received queries and $P_{rnd}$ increases. If nodes are finally broadcasting queries to all neighbors, even the exact test fails, as $P_{rnd}(A)^{-1} = P_{rnd}(B) = 1$.

Thus, the attack becomes more difficult with lower load. If the load on the network is low, the attacker has to increase the load for the attack to be successful. Note that nodes with low load are accepting zero priority queries and no credit is required to increase the load.

**Number of Queries Required.** To estimate how many messages the initiator can send without being identified we assume $P_{rnd}$ to be equal for all nodes in GNUnet. Let $l$ be the distance of the attacker to the initiator.

1. The attacker will receive linkable queries from intermediary node $i$ with probability $P^i_{rnd}$, where $1 \leq i \leq l$. The first query is therefore received after the issuer has sent $m_i \geq 1/P^i_{rnd}$ queries.
2. The attacker will receive linkable queries from the neighbors of the initiator with probability $P^2_{rnd}$. The first query is therefore received after the issuer has sent $m_0 = 1/P^2_{rnd}$ queries.

Altogether, a lower bound on the total number of queries required to determine the initiator is

$$m \geq \frac{1}{P^2_{rnd}} + \sum_{i=1}^{l} \frac{1}{P^i_{rnd}}$$

### 4.3 Hot Path Routing

With hot path routing the nodes to receive a query are selected by the number of their valid responses to recent queries. Neighbors that respond more often are more likely to receive further queries. The probability that node $A$ selects its neighbor $I$ is $P_{hot}(A, I) = r/q$, where $q$ is the number of queries $A$ has recently (in a time interval) sent to $I$ and $r$ is the number of valid responses. Thus, if the attacker is a hot node, the shortcut attack becomes considerably easier:

1. More queries are routed to the attacker and thus fewer queries are required in total. If a node is on the hot path, then $P_{hot} > P_{rnd}$, otherwise $P_{hot} \approx P_{rnd}$.
2. Due to the random routing, the hot path still leaks queries to neighbor nodes that are not on the hot path.

Both properties help the attacker to determine the initiator node. Figure 3 shows a hot path, where the attacker is a hot node. In this case, the attacker can efficiently test, which neighbor of $A$ is closer to the initiator.

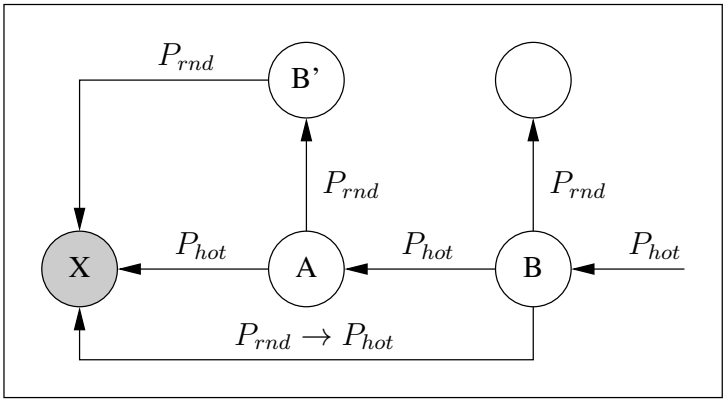



**Fig. 3.** Attacking Hot Paths.

**Testing.** To test which of the neighbor nodes $B_i$ of the current preceding node $A$ is closer to the initiator, we compare the number of queries received from those nodes. Let $q_A$ be the number of linkable queries the attacker $X$ receives from $A$. Then the attacker can expect to receive the following number of linkable queries from a tested node $B$ in the same time interval. If the tested node $B$ is closer to the initiator, the attacker expects to receive

$$q_B = q_A \cdot \frac{P_{rnd}(B)}{P_{hot}(A,X) \cdot P_{hot}(B,A)} \approx q_A \cdot P_{rnd}/P_{hot}^2$$

linkable queries[2] from this node. Otherwise, the tested node is more distant from the initiator and the attacker expects to receive only

$$q'_B = q_A \cdot \frac{P_{rnd}(A) \cdot P_{rnd}(B)}{P_{hot}(A,X)} \approx q_A \cdot P_{rnd}^2/P_{hot}$$

linkable queries from this node.

It is important that $P_{rnd}/P_{hot}^2 \gg P_{rnd}^2/P_{hot}$ to succeed with the simplified test. As $P_{hot} \geq P_{rnd}$ and $P_{rnd}$ is small, the node that is closer to the initiator can be determined efficiently.

**Number of Queries Required.** To estimate how many messages the initiator can send without being identified we assume $P_{rnd}$ and $P_{hot}$ to be equal for all nodes in GNUnet. Let $l$ be the length of the hot path.

1. The attacker will receive linkable queries from the preceding node with probability $P_{hot}^l$. The first query is therefore received after the issuer has sent $m_l \geq 1/P_{hot}^l$ queries.

[2] Moreover, as shown in Figure 3 the local receiving probability will increase over time to $P_{hot}$, as $B$ learns that the attacker is a hot node, while $A$ may be unable to respond to $B$'s queries without the help of the attacker.

2. The attacker will receive linkable queries from intermediary node $i$ on the hot path with probability $P_{hot}^{i-1} P_{rnd}$, where $1 \leq i \leq l-1$. The first query is therefore received after the issuer has sent $m_i \geq 1/P_{hot}^{i-1} P_{rnd}$ queries.
3. The attacker will receive linkable queries from the neighbors of the initiator with probability $P_{rnd}^2$. The first query is therefore received after the issuer has sent $m_0 = 1/P_{rnd}^2$ queries.

Altogether, a lower bound on the total number of queries required to determine the initiator is

$$m \geq \frac{1}{P_{hot}^l} + \frac{1}{P_{rnd}^2} + \frac{1}{P_{rnd}} \cdot \sum_{i=0}^{l-2} \frac{1}{P_{hot}^i}$$

### 4.4 Discussion

In the previous sections we have presented lower bounds for the number of queries required to determine the initiator. Those lower bounds are only a rough estimation under several simplifications (e.g. no TTLs, no loops, etc.). Nevertheless, we now try to discuss the efficiency of the shortcut attack in practice. Therefore, we assume "realistic" parameters: the initial distance of the attacker to the initiator is $l = 6$ and every node in the network uses $P_{hot} = 0.9$ and $P_{rnd} = 0.1$.

If random routing is only used, GNUnet seems to be secure, as the attacker is relatively far away from the initiator and receives every query sent by the initiator only with probability $10^{-6}$. Thus, it is very unlikely that the attacker even notices a download. The question is however whether the shortest path from the initiator to the attacker is indeed relatively long.

Let $n$ be the total number of nodes, $c$ be the number of attackers, and $k$ be the number of neighbors. Assuming that each node chooses its neighbors randomly, then the probability that a node connects to $i$ attackers is

$$p_i = \frac{\binom{c}{i}\binom{n-c}{k-i}}{\binom{n}{k}}$$

For $k \ll n$ the probability that a node is not connected to an attacker can be simplified to $p_0 = (1 - c/n)^k$. Thus, unless the fraction $c/n$ is negligible, the number of neighbors has a great impact on the length of the shortest path to an attacker. Interestingly, having fewer neighbors not only increases the length of the path, but also increases $P_{rnd}$, which has two effects: On one side, it becoms more difficult for the attacker to test neighbor nodes, but on the other side more queries are routed to the attacker, which helps the attacker unless $P_{rnd} = 1$.

Therefore, it seems to be important that each node does not choose its neighbors randomly, but uses a trust-metric, e.g. [LA98], to minimize the risk connecting to an attacker. Furthermore, such a trust-metric makes shortcut attacks much more unlikely, as a node can reject connection requests from untrusted nodes.

While the security of random routing remains controversial, our shortcut attack is very successful, if the attacker is able to exploit a hot path. With the parameters of the example above only 82 multiqueries are required to determine the initiator and thus downloading a file of approximately 2 MB size is potentially dangerous.

Even if an implementation of the attack actually requires a multiple of the estimated number of queries, it shows that hot path routing is a potential weakness. The best strategy for an attacker is to provide GNUnet with several high bandwith nodes. As hot paths always leak some queries to the neighbor nodes, the attacker can very efficiently test which nodes are on the hot path, until the initiator is discovered.

To summarize, hot path routing not only improves GNUnet's performance, but it also increases the efficiency of our attack. As GNUnet is very inefficient without hot path routing, we suggest to make hot paths more secure.

- To prevent the improved shortcut attack, hot paths must not leak linkable queries to nodes not on the hot path. Therefore, hot paths have to be much more static, and all nodes on the hot path must have knowledge which queries are linkable.
- But even with very static paths, the basic shortcut attack can be used to discover the initiator. While the basic shortcut attack requires credit, it should be noted that the attacker earns this credit by responding to queries.

Therefore, hot paths should be very static, i.e. shortcuts should not be allowed on hot paths.

## 5 Censoring GNUnet

Assuming that GNUnet turns out to be practical and is in wide use, we expect that there will be attempts to censor GNUnet. We present two methods to censor GNUnet: rubber-hose cryptanalysis and content filtering.

### 5.1 Rubber-Hose Cryptanalysis

One possibility for censorship is to apply "Rubber-Hose Cryptanalysis" by forcing the owner of a node to remove certain content. Rubber-hose cryptanalysis can only be used to censor infrequently requested content that is not widely distributed, as it is necessary to discover which nodes store the content to be censored. On the other side GNUnet also has built-in censoring capabilities as infrequently requested content is automatically removed from the node's caches and rubber-hose cryptanalysis becomes more or less unnecessary. To prevent infrequently requested content from vanishing, GNUnet provides an alternative method to add content.

- The default method to add content is inserting: All blocks are generated and stored in the local cache in encrypted form.

– Alternatively, indexing can be used to add content: Only the RBlock is generated and all other blocks are produced on the fly upon request.

While indexing saves space and prevents infrequently requested content from vanishing from GNUnet, this approach has a major drawback: A reverse shortcut attack can be used to discover nodes providing such infrequently requested indexed content.

**Reverse Shortcut Attack.** The shortcut attack can similarly be used to discover the responding node by exchanging the roles of $A$ and $X$ (see Figure 2). The attacker can successively get closer to the responding node by eliminating intermediary nodes one after the other.

Compared to the basic shortcut attack, where it is quite difficult for an attacker to force a node to reveal the preceding node, the reverse shortcut attack is much simpler.

1. The attacker floods $B$ with high priority queries. Due to the promised credit $B$ will try to respond to those queries. The attacker may hide his attack by using several malicious nodes to flood the attacked node.
2. The load on the attacked node $B$ increases and the node decides not to indirect but to forward (or to drop) some traffic. When forwarding queries $B$ does not overwrite $A$'s return address, so that the $X$ will send responses directly to the attacker.
3. After the attacker has received a valid response from $X$, the attacker will send all further queries to $X$.

Thus, the path is very static, which makes this attack practical. While the attack still requires credit to flood the node, it should be noted that the attacker is downloading the content, therefore, the attacker can postpone queries until he has acquired enough credit at the attacked node to proceed with the attack.

### 5.2 Content Filtering with Licenses

Rubber-hose cryptanalysis is neither suited for large scale censorship nor to censor frequently requested content that is stored in many locations. Therefore, we present a content filtering mechanism that can be legally enforced to make sharing illegal content nearly impossible.

Our filter mechanism makes use of the fact that I- and DBlocks are stored under the hash value of their encrypted content. Thus, indexing illegal content is possible and censorship can be applied very efficiently by allowing nodes to only deliver root IBlocks together with a valid license (in principle it is sufficient to filter root IBlocks, however, this requires that root IBlocks are distinguishable from other blocks). A censoring authority issues licenses by signing the query-hash $H(E_{H(I)}(I))$ of the root IBlock. There are two types of licenses, positive and negative licenses. A positive license allows, a negative license prohibits delivering the corresponding content.

**Positive License:** A positive license certifies that $H(E_{H(I)}(I))$ is currently not on the index. Positive licenses are always time restricted. Thus, to issue a positive license, the censoring authority need not check the actual content.

**Negative License:** A negative license certifies that $H(E_{H(I)}(I))$ is on the index. Negative licenses are not time restricted.

A node that wants to respond to a query, but does not possess a positive license for the IBlock (e.g. the content is new or the positive license is not valid anymore), first tries to retrieve a license from GNUnet. If no license is available, it directly requests a license from the censoring authority. Thus, for frequently requested legal content the permission to deliver this content is widely distributed in GNUnet and can be retrieved efficiently.

**Censoring.** Searchability and censorship-resistance exclude each other to a certain extent. If content is easy to find for users, it is similarly easy for others to find illegal content:

1. Content owners can search for their copyrighted content. After finding such content, the corresponding root IBlock is sent to the censoring authority together with a proof of ownership.
2. The censorship authority itself searches for illegal content and issues negative licenses for all found IBlocks.

To check whether a node applies content filtering, the censoring authority only has to search for indexed content. The owner of a node that returns such content (without a valid positive license) may be prosecuted for e.g. copyright infringement. As every owner of a node can be prosecuted, GNUnet may discourage users to share illegal content even more than any non-anonymous approach.

**Multiple Censoring Authorities.** Instead of using only a single censoring authority, it is also possible to use multiple censoring authorities, depending on the jurisdiction of the node indirecting the content.

### 5.3 Discussion

Constructing an anonymous, censorship-resistant network is very difficult. Besides Free Haven [DFM01] the only approach achieving a similar goal is sketched in [Ser02]. There, it is also discussed why most censorship-resistant networks (e.g. Publius [MWC00], Freenet [CSWH01], and Tangler [WM01]) fail to resist rubber-hose cryptanalysis. Unfortunately, due to indexing GNUnet is even more affected by rubber-hose cryptanalysis than those approaches. If indexing is used to add infrequently requested content to GNUnet, the node risks being identified with the reverse shortcut attack. Therefore, indexing should not be used for such content.

While the reverse shortcut attack can only be used to censor infrequently requested indexed content, our filter mechanism is well suited to censor popular

content. Content filtering is a very strong, controversial attack, as it requires the nodes to apply the filter. Content filtering is also often compared to shutting down GNUnet. However, we think that is important to consider such attacks:

- History has shown that such attacks are used in practice, e.g. Napster was obliged to filter content before it was shut down.
- Content filtering can be applied very efficiently with only minor changes in GNUnet, resulting in an anonymous but "clean" network.

As every block in GNUnet is identified by an unique identifier, content filtering is always possible. Removing unique identifiers, also removes the ability to automatically replicate content. Therefore, preventing content filtering seems to be impossible.

## 6 Conclusion

We have pointed out some potential weaknesses of GNUnet. A problem that is inherent to the design of GNUnet is the decision to split larger files into many small linkable pieces. To download a file all those linkable pieces have to be retrieved separately. An attacker who can link those related queries is able to determine the initiator of the download. Depending on the distance of the attacker and on the load of the network a huge number of queries may be necessary to determine the initiator. However, to be more efficient GNUnet tries to automatically optimize routing with hot paths. We have shown that the attacker can benefit from hot paths, which makes our attack much more efficient.

Another problem that we have discussed is censorship-resistance. We have shown that GNUnet is vulnerable to rubber-hose cryptanalysis, but we additionally have presented an efficient content filter for GNUnet. If content filtering is legally enforced, censoring GNUnet is possible at a very large scale.

## Acknowledgements

Many thanks to Christian Grothoff from GNUnet for discussing the attacks and to Andrei Serjantov for commenting on earlier versions of this paper.

## References

And96. Ross J. Anderson. The eternity service. In *Proceedings of Pragocrypt '96*, 1996.

BG03. Krista Bennett and Christian Grothoff. GAP – practical anonymous networking. In *Designing Privacy Enhancing Technologies – International Workshop on Design Issues in Anonymity and Unobservability 2003*. Springer-Verlag, 2003.

BGHP02. Krista Bennett, Christian Grothoff, Tzvetan Horozov, and Ioana Patrascu. Efficient sharing of encrypted data. In *Information Security and Privacy – 7th Autralasian Conference (ACISP 2002)*, Lecture Notes in Computer Science 2384, pages 107–120. Springer-Verlag, 2002.

BPS01. Oliver Berthold, Andreas Pfitzmann, and Ronny Standtke. The disadvantages of free MIX routes and how to overcome them. In *Designing Privacy Enhancing Technologies – International Workshop on Design Issues in Anonymity and Unobservability 2000*, Lecture Notes in Computer Science 2009, pages 30–45. Springer-Verlag, 2001.

Cha81. David Chaum. Untraceable electronic mail, return adresses and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

CMH+02. Ian Clarke, Scott G. Miller, Theodore W. Hong, Oskar Sandberg, and Brandon Wiley. Protecting free expression online with Freenet. *IEEE Internet Computing*, pages 40–49, 2002.

CSWH01. Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Designing Privacy Enhancing Technologies – International Workshop on Design Issues in Anonymity and Unobservability 2000*, Lecture Notes in Computer Science 2009, pages 46–66. Springer-Verlag, 2001.

DFM01. Roger Dingledine, Michael J. Freedman, and David Molnar. The free haven project: Distributed anonymous storage service. In *Designing Privacy Enhancing Technologies – International Workshop on Design Issues in Anonymity and Unobservability 2000*, Lecture Notes in Computer Science 2009, pages 67–95. Springer-Verlag, 2001.

FP00. Justin Frankel and Tom Pepper. Gnutella v. 0.56. Nullsoft, 2000.

GPB+02. Christian Grothoff, Ioana Patrascu, Krista Bennett, Tiberiu Stef, and Tzvetan Horozov. GNET. Whitepaper version 0.5.2, 2002.

Gro03. Christian Grothoff. An excess-based economic model for resource allocation in peer-to-peer networks. *Wirtschaftsinformatik*, (3), 2003.

LA98. Raph Levien and Alex Aiken. Attack-resistant trust metrics for public key certification. In *Proceedings of the 7th USENIX Security Symposium*, 1998.

MWC00. Aviel D. Rubin Marc Waldman and Lorrie Faith Cranor. Publius: A robust, tamper-evident, censorship-resistant, web publishing system. In *Proc. 9th USENIX Security Symposium*, pages 59–72, 2000.

PK01. Andreas Pfitzmann and Marit Köhntopp. Anonymity, unobservability, and pseudonymity – a proposal for terminology. In *Designing Privacy Enhancing Technologies – International Workshop on Design Issues in Anonymity and Unobservability 2000*, Lecture Notes in Computer Science 2009, pages 1–9. Springer-Verlag, 2001.

RR98. Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

Ser02. Andrei Serjantov. Anonymizing censorship resistant systems. In *Peer-to-Peer Systems, First International Workshop – IPTPS 2002*, Lecture Notes in Computer Science 2429, pages 111–120. Springer-Verlag, 2002.

Shm02. Vitaly Shmatikov. Probabilistic analysis of anonymity. In *15th IEEE Computer Security Foundations Workshop*, pages 119–128. IEEE Computer Society Press, 2002.

WALS02. Matthew Wright, Micah Adler, Brian N. Levine, and Clay Shields. An analysis of the degradation of anonymous protocols. In *Network and Distributed System Security Symposium – NDSS 2002*. Internet Society, 2002.

WM01. Marc Waldman and David David Mazières. Tangler: A censorship-resistant publishing system based on document entanglements. In *ACM Conference on Computer and Communications Security*, pages 126–135. ACM Press, 2001.

# A Component Architecture for Dynamically Managing Privacy Constraints in Personalized Web-Based Systems

Alfred Kobsa

School of Information and Computer Science
University of California, Irvine
http://www.ics.uci.edu/~kobsa

**Abstract.** User-adaptive (or "personalized") systems on the web cater their interaction to each individual user and provide considerable benefits to both users and web vendors. These systems pose privacy problems, however, since they must collect large amounts of personal information to be able to adapt to users, and often do this in a rather inconspicuous manner. The interaction with personalized systems is therefore likely to be affected by users' privacy concerns, and is in many cases also subject to privacy laws and self-regulatory privacy principles. An analysis of nearly 30 international privacy laws revealed that many of them impose severe restrictions not only on the data that may be collected but also on the personalization *methods* that may be employed. For many personalization goals, more than one methods can be used that differ in their data and privacy requirements and their anticipated accuracy and reliability. This paper presents a software architecture that encapsulates the different personalization methods in individual components and, at any point during runtime, ascertains the dynamic selection of the component with the optimal anticipated personalization effects among those that are permissible under the currently prevailing privacy constraints.

## 1 Personalized Systems on the Web: Benefits and Methods

User-adaptive (or "personalized") computer systems take individual characteristics of their current users into account and adapt their behavior accordingly. Such systems have already been deployed in several areas, including education and training (e.g., [1]), online help for complex PC software (e.g., [2, 3]), dynamic information delivery (e.g., [4]), provision of computer access to people with disabilities (e.g., [5, 6]), and to some extent information retrieval. In several of these areas, benefits for users could be empirically demonstrated.

Since about 1998, personalization technology is being deployed to the World Wide Web where it is mostly used for customer relationship management. The aim is to provide value to customers by serving them as individuals and by offering them a unique personal relationship with the business (the terms *micro marketing* and *one-to-one marketing* are being used to describe this business model [7, 8]). Current person-

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 177–188, 2003.

alization on the web is still relatively simple. Examples include customized content provision (e.g., personalized information on investment opportunities, or personalized news), customized recommendations or advertisements based on past purchase behavior, customized (preferred) pricing, tailored email alerts, and express transactions [9]. Personalization that is likely to be found on the web in the future includes, e.g.,

- product descriptions whose complexity is geared towards the presumed level of user expertise;
- tailored presentations that take users' preferences regarding product presentation and media types (text, graphics, video) into account;
- recommendations that are based on recognized interests and goals of the user; and
- information and recommendations by portable devices that consider the user's location and habits.

A number of studies indicate that users seem to find personalization on the web useful [10, 11], and that they stay longer at personalized websites and visit more pages [12]. Other research demonstrates that personalization also benefits web vendors with respect to the conversion of visitors into buyers [13], "cross-selling" [14], and customer retention and development [15, 16].

Personalized systems utilize numerous techniques for making assumptions about users, such as domain-based inference rules, stereotype techniques, machine learning techniques (e.g content-based filtering, and clique-based or "collaborative" filtering), plan recognition methods, logic-based reasoning, Bayesian inferences, and many more (see [17] for a recent survey). These techniques have different requirements regarding the data that must be available. For instance, most machine learning techniques assume that a large number of raw data (such as a user's clickstream data) is available and that all learning is performed at one time. Since individual sessions are often too short to deliver sufficient data about a user, these techniques are therefore typically applied to data from several sessions with the user. In contrast, incremental techniques can learn in several steps, taking the new raw data of the current session and the previous learning results into account.

## 2 Privacy Problems Caused by Personalized Systems

Personalized systems generally operate in a data-driven manner: more personalization can be performed the more data about the user is available, and personalization based on more data will also tend to be more accurate and more individualized. User-adaptive systems therefore collect considerable amounts of personal data and "lay them in stock" for possible future usage. Moreover, the collection of information about users is often performed in a relatively inconspicuous manner (such as by watching their web navigation behaviors). Personalized systems are therefore most certainly affected by the privacy concerns that a majority of today's Internet users articulates, by privacy laws that are in place, and by company and sector privacy policies.

### 2.1 Users' Privacy Concerns

Numerous consumer surveys have been conducted so far that consistently reveal widespread privacy concerns among today's Internet users[1]. Respondents reported being (very) concerned about, e.g., threats to their privacy when using the Internet (81 - 87%), about divulging personal information online (67 - 74%), and about being tracked online (54 - 77%). They indicated leaving web sites that required registration information (41%) having entered fake registration information (24 - 40%), and having refrained from shopping online due to privacy concerns or having bought less (24 - 32%). An analysis of results from thirty surveys with a focus on web personalization is given in [18].

Hardly any survey data exists on whether Internet users will agree with the usage of their personal data for personalized interaction. In a poll by an industry advocacy group for web personalization [11], 51% of the respondents indicated to be willing to give out information about themselves in order to receive an "online experience truly personalized for them" (the subjects of this study were however recruited from a "permission-based opt-in list" which may have biased the sample). It seems prudent to assume that the general Internet privacy concerns that were documented by the mentioned consumer surveys also apply to the usage of personal data for web personalization purposes. Caution must be exercised however since users who claim having privacy concerns do not necessarily exhibit a more privacy-minded interaction with web sites, as was demonstrated in experiments by [19].

### 2.2 Privacy Laws and Self-regulatory Privacy Principles

Privacy laws protect the data of identified or *identifiable* individuals. For privacy laws to be applicable, it is thus not required that the system actually identifies the user, but only that it is *possible* to identify the user with reasonable efforts based on the data that the system collects. The latter situation often applies to personalized systems. The privacy laws of many countries not only regulate the processing of personal data in the national territory, but also restrict the trans-border flow of personal data, or even extend their scope beyond the national boundaries. Such laws then also affect personalized web sites abroad that serve users in these regulated countries, even when there is no privacy law in place in the jurisdictions in which these sites are located.

We collected nearly 30 international privacy laws and categorized them by criteria that affect the design of personalized systems the most [20]. Categories include registration duties, record-keeping duties, reporting duties, disclosure duties at the website, duty to respect certain user requests, duty to respect user vetoes ("opt out"), duty to ask for user permission ("opt in"), exceptions for very sensitive data, restrictions on data transfer abroad, restrictions on foreign sites collecting data inland, archiving/destruction of personal data, and "other" impacts on personalization. We found that if privacy laws apply to a personalized website, they often not only affect the

[1] Links to most surveys that are available online can be found at http://www.privacyexchange.org/iss/surveys/surveys.html.

conditions under which personal data may be collected and the rights that data subects have with respect to their data, but also the *method*s that may be used for processing them. Below is a sample of several legal restrictions that substantially affect the internal operation of personalized hypermedia applications (more constraints will be discussed in the application example).

- *Usage logs must be deleted after each session,* except for billing purposes and certain record-keeping and fraud-related debt recovery purposes [21]. This provision affects, e.g., the above-mentioned machine learning methods that can be employed in a personalized hypermedia system. If learning takes place over several sessions, only incremental methods can be employed since the raw usage data from previous sessions have all to be discarded.
- *Usage logs of different services may not be combined, except for accounting purposes* [21]. This is a severe restriction for so-called central user modeling servers that collect user data from, and make them available to, different user-adaptive applications [22].
- *User profiles are permissible only if pseudonyms are used. Profiles retrievable under pseudonyms may not be combined with data relating to the bearer of the pseudonym* [21]. This clause mandates a Chinese wall between the component that receives data from identifiable users, and the user modeling component which makes generalizations about pseudonymous users and adapts hypermedia pages accordingly.
- *No fully automated individual decisions are allowed* that produce legal effects concerning the data subject or significantly affect him, and which are based solely on automated processing of data intended to evaluate certain personal aspects relating to him, such as his performance at work, creditworthiness, reliability, conduct, etc [23]. This prohibition has impacts on learner-adaptive hypermedia systems for tutoring [24]. E.g., if such systems assign formal grades, there has to be a human in the loop somewhere.
- *Anonymous or pseudonymous access and payment must be offered if technically possible and reasonable* [21, 25].
- *Strong encryption software is regulated in France* [26], which may have impacts on the use of encryption to protect user data in transit when a personalized website or the user is located in France.

In addition to legislative regulations, privacy practices of personalized web sites are also restricted by self-regulatory privacy principles, such as company-specific privacy policies or sector-specific principles (e.g., [27]). These principles can also severely impact the permissibility of personalization methods.

## 3 Privacy Management

### 3.1 Pseudonymous and Identified Interaction

Two principled solutions are possible to cater to privacy requirements in personalized systems. One direction is to allow users to remain anonymous with regard to the per-

sonalized system (and possibly even the whole network infrastructure) whilst enabling it to still link the same user across different sessions, so that it can cater to her individually. In [28, 29], we present a reference model for pseudonymous interaction between users and web-based applications in which full personalization can nevertheless take place[2].

Pseudonymous interaction seems to be appreciated by users, even though only a single user poll addressed this question explicitly so far [30]. One can expect that anonymity will encourage users to be more open when interacting with a personalized system, thus facilitating and improving the adaptation to this user. The fact that in most cases privacy laws do not apply any more when interaction is anonymous also relieves the application provider from restrictions and duties imposed by such laws. Finally, anonymous and pseudonymous interaction are sometimes even legally mandated if they can be realized with reasonable effort [21, 25].

Anonymous and pseudonymous interaction also has several drawbacks though: it requires an elaborate anonymity infrastructure, it is currently difficult to preserve when payments, physical goods and non-electronic services are being exchanged, and it harbors the risk of misuse. Anonymous personalization is also restricted to electronic channels only. Pseudonymous data cannot be used for cross-channel communication (sending a brochure to a web customer by mail) and cross-channel recognition (recognizing a web customer in a brick and mortar store). These drawbacks become increasingly important since the number of web-only vendors is constantly shrinking.

In the second principled approach to rejoining personalization and privacy, the user would not remain anonymous. Privacy issues are taken into account by respecting privacy laws, self-regulatory privacy principles, and/or users' privacy preferences. This paper deals exclusively with this second approach. It is specifically concerned with architectural issues of privacy management in personalized systems, i.e. software architectures and processes that allow a personalized system to dynamically cater to user's privacy wishes and to regulatory constraints.

### 3.2 Current Work on Privacy Management

Current work in privacy management is mostly concerned with the specification of privacy constraints for data, relating these constraints to software and business processes, and enforcing privacy constraints automatically. [31] introduces privacy metadata tables which indicate the external recipients and the retention period, for each usage purpose and for each piece of information (attribute) collected for that purpose. A second meta-table specifies access permissions. Processes like the Privacy Constraint Validator, the Attribute Access Control and the Data Retention Manager check the compliance with privacy preferences and privacy policies.

IBM's Enterprise Privacy Architecture [32, 33] maps customer preferences and data onto business processes, privacy rules, technology and the enterprise architecture

[2] This model also protects the anonymity of the central user modeling server that contains the user's data since knowledge about its location may reveal the identity of the user, e.g. when it is hosted in the user's local network.

as a whole, and thereby provides a mechanism for analyzing business processes in a privacy context. A "technical reference model" helps guarantee privacy at the transactional level. This model relies on object, data and rules models to build applications that support and enhance privacy and collectively determine what privacy-relevant data is collected and how it must be handled. An authorization director evaluates the given policies and decides whether or not access requests to data sources are granted. [34] focuses on the formulation of enterprise-independent privacy policies in the E-P3P Privacy Policy Language to express and enforce access restrictions to personal data in legacy systems. In a similar vein, [35] study the more expressive logic-based Authorization Specification Language.

[36] presents a formal security model based on state machines, to enforce legal privacy requirements (such as purpose binding or necessity of data processing). The model is based on the integrity concepts of well-formed transactions and separation of duty.

This work complements existing approaches in that it focuses on ways in which a personalized system can dynamically adjust to the currently prevailing privacy constraints. Numerous stipulations in privacy laws, and most likely also user privacy concerns, influence personalization methods in very different ways. A global preformulated policy for the selection of personalization methods under each different combination of impact factors does not seem feasible. Instead, a set of personalization methods must be dynamically selected at runtime, considering the current privacy constraints within the general goal of maximizing personalization benefits. In the remainder of this paper, we will discuss an architecture that utilizes functionally related software components for this purpose.

## 4 Redundant-Component Architectures

### 4.1 Overview

Component architectures have been widely advocated as a means for flexibly assembling software objects both at design time and at run time (e.g., [37, 38]). A *redundant component array* (RAIC) [39-41] is a group of similar or identical components. It uses the services from one or more components inside the group to provide services to applications. Applications connect to a RAIC and use it as a single component. They typically do not know the individual components that underlie a RAIC. Component membership in a RAIC can be *static* or *dynamic*. Components in a static RAIC are explicitly assigned at design time whereas components in a dynamic RAIC may still be incorporated during run-time.

Depending on the types and relations of components in a RAIC, it can be used for many different purposes such as providing higher reliability, better performance, or greater flexibility than what could be achieved by a single component alone. In this paper, we will restrict ourselves to those aspects of RAICs that are relevant for personalization purposes.

Three major types of relations govern the relationship between components in RAICs:

**Interface Relations:** Interfaces determine the way in which applications interact with components. Components $A$ and $B$ have *inclusionary* interfaces (abbreviated $A \subseteq_I B$) if and only if every possible call to each function in the interface that $A$ implements can be converted to a call to a corresponding function in $B$'s interface without loss of information. Stricter kinds of interface relations are *identical* and *equivalent;* other types are *similar* and *incomparable* (see [39] for their definitions).

**Domain Relations:** The domain of a component refers to the scope in which it can provide service, i.e. the range of its input data. Two components $A$ and $B$ are said to have *inclusionary* domains (abbreviated $A \subseteq_{\mathcal{D}} B$) if and only if $A$'s domain is a subset of $B$'s domain, i.e. each input in $A$'s valid input domain is also in $B$'s valid input domain. A stricter kind of domain relation is *identical;* other types are *exclusionary* and *incomparable*.

**Functional Relations:** Functional relations refer to the functionality of components. Two types are relevant for our purposes:

*Similar*: Two components have similar functionalities (abbreviated $A \approx_{\mathcal{F}} B$) if they are designed to perform the same tasks but possibly with different requirements (e.g., with different accuracy).

*Inclusionary*: Two components $A$ and $B$ have inclusionary functionality (abbreviated $A \subseteq_{\mathcal{F}} B$) if and only if the functionalities of component $A$ form a subset of those of $B$ (i.e., if every possible task that $A$ performs can be carried out by $B$, possibly with different accuracy).

Inclusionary functionality is obviously more general than functional similarity. Even stricter relations are functional *equivalence* and *identity*, but they are not relevant for our purposes.

Relations between two components in RAICs can be "manually" identified, by analyzing their interfaces, service domains and functionality. [39] discusses methods to also determine the relations between components automatically. For certain types of analysis, type information is required that is currently not generally available (such as the "reflection" information on the .NET platform for the analysis of interface relations, or a typology of functionality for the analysis of the functional relations).

A *RAIC controller*, among other things, determines which component(s) inside the RAIC should deliver the services that are offered by the RAIC. For our purposes, the decision is based on a partial order $<$ between components, the so-called *activation preference*. $A < B$ denotes that services to the application should be delivered by $B$ rather than $A$ if both are in principle eligible. The relationship between two components in this order can be determined empirically ("when $A$ and $B$ could both deliver a certain service, which of them should be preferred?"). In many domains (e.g., personalization), an approximation of the activation preference can also be computed based on component relations.

### 4.2 Redundant Personalization Components

The central tenets of this work are:

1. A personalized system must dynamically cater to changing privacy concerns of users during runtime, and to privacy laws that are in effect in the jurisdictions of both the user and the data processor (and possibly other jurisdictions as well if part of the personal data is located or processed therein).
2. User preferences and privacy laws may have an impact on both the usable data and the permissible methods.
3. A personalized system can dynamically cater to these (changing) requirements when it is designed in a RAIC-like architecture, where RAICs contain functionally inclusionary or at least similar components. At any given time, the services of the RAIC are delivered by that component that is both ranked highest in the activation preference order and meets all current privacy requirements. If a component cannot operate any more due to a change in the privacy requirements, a substitute component is selected based on the activation preference order.

The activation preference order depends on the application domain. For personalization purposes, an approximation of this relation can be constructed based on component relations, using the following rules:

(T 1) If $A \subseteq_{\mathcal{F}} B$ and $A \subseteq_{I} B$, then $A < B$

(i.e., if B's functionality and interface includes A's functionality and interface, then B should be preferred)

(T 2) If $A \approx_{\mathcal{F}} B$ and $A \subseteq_{I} B$ and $A \subseteq_{\mathcal{D}} B$, then $A < B$

(i.e., if $A$ and $B$ are functionally at least similar, and $B$ uses more data than $A$ and includes $A$'s interface, then $B$ should be preferred since it will probably deliver higher-quality results).

### 4.3 An Example in a Personalized Recommender Domain

We will illustrate our approach using the example of a web store that gives personalized purchase recommendations to web visitors by predicting items in which the user is presumably interested. The service 'predicting the user's interest' is delivered by a RAIC that contains five different components. These components generate predictions based on different data, and use different methods for this purpose (see [17] for a more comprehensive survey of interest prediction methods):

Component A: makes predictions based on the user's demographic data (age, gender, profession, ZIP), by drawing conclusions based on market segmentation data;

Component B: makes predictions based on the user's page visits (during the current session only), using "quick" one-time machine learning methods;

Component C: makes predictions based on the user's demographic data and page visits (in the current session only), using a combination of the methods in A and B;

Component D: makes predictions based on the user's page visits during several sessions, using incremental machine learning methods (the user trace is thereby stored between sessions)

Component E: makes predictions based on the user's demographic data and her page visits during several sessions, using a combination of the methods in A and D (the user trace is again stored between sessions).

Through domain analysis at design time, we can determine that

(1) $A \approx_{\mathcal{F}} B \approx_{\mathcal{F}} C \approx_{\mathcal{F}} D \approx_{\mathcal{F}} E$

In the future, (1) may be inferable by meta-descriptions included in every personalization component that locates the component in a function taxonomy.

The following additional relations can all be automatically determined at design time, possibly with the help of limited meta information [39]:

(2) $A \subseteq_I C$, $B \subseteq_I C$, $B \subseteq_I D$, $C \subseteq_I E$, $A \subseteq_I E$, $D \subseteq_I E$

(3) $A \subseteq_{\mathcal{D}} C$, $B \subseteq_{\mathcal{D}} C$, $B \subseteq_{\mathcal{D}} D$, $C \subseteq_{\mathcal{D}} E$, $A \subseteq_{\mathcal{D}} E$, $D \subseteq_{\mathcal{D}} E$

With (T 2) we can now conclude that

(4) $A < C$, $B < C < E$, $D < E$

Based on this partial activation preference order, the RAIC controller can determine that E should be used with highest priority, and that C or D should be used as substitutes if E does not meet the current privacy constraints. (4) is however only an automatically generated approximation of <. Additional preferences may be entered based on domain knowledge (such as that $A < B$, with the – empirically unproven – rationale that interest predictions based on users' individual web navigation outperform predictions based on users' demographic profiles).

Components A – E have numerous prerequisites for their operation, which may change during runtime and therefore have to be continuously verified:

1. *Availability of data*: A will not be able to operate if the user did not provide the necessary demographic data. B and D will not be able to operate during the first few interactions with a new user.
2. *Privacy laws*: In many jurisdictions (e.g., in all EU member states [23]), components A - E may only operate if the user has unambiguously given her consent to the processing of her personal data for the purpose of personalized interaction. Even when such a general consent was given, the user still has the right to specifically opt out of B if the web store is located in Germany [21, 25]. C and E are illegal for German web stores without the user's consent since use profiles may only be constructed pseudonymously and may not be combined with data of the bearer of the pseudonym. D is illegal in Germany without the user's consent since use data of online services may only be stored beyond a user session for billing purposes and certain record-keeping and fraud-related fee recovery purposes.
3. *Self-regulatory privacy principles*. If the web store is a signatory of the U.S. Network Advertisers Initiative, C and E may not operate unless the user has consented to the merger of non-personally identifiable use data and demographic data (if the latter is personally identifiable [27]).

4. *Users' individual privacy preferences*: B - E should not operate if the user communicates to the web store that he "does not like being watched while browsing the web store".

A considerable amount of work already exists on how to communicate users' privacy preferences [42], how to formalize textual privacy policies [34, 35, 43], and how to compare policy requirements with permissions that were given by the users [31, 33]. A decision about which of the components A - E is allowed to operate at a given time can be made using any of these methods, and we will therefore not deal with this issue here. The RAIC can use (4) at any given time to determine which of the permissible personalization components *should* operate since this component provides optimal personalization under the given privacy constraints.

## 5 Conclusion

While personalization on the web is demonstrably beneficial for both web users and web vendors, privacy issues pose a severe obstacle to its broad dissemination. If the user's identity is known to the system or if the user is identifiable, personalization is subject to privacy laws, self-regulatory privacy principles and individual user concerns. These constraints not only affect the kinds of data that may be used for personalization purposes, but also the admissibility of the numerous personalization methods that have been developed to date.

This paper discussed a software architecture in which personalization methods are individually embodied in software components, and where components with similar functionality but different data and privacy requirements are placed into groups (the so-called RAICs) that offer services to applications collectively. Applications that utilize services of a RAIC are unaware of its internal structure, and of the component that currently provides these services. An activation preference order instructs a RAIC controller which component should preferably deliver these services if more than one component meet the current privacy requirements. This architecture allows for a flexible dynamic adjustment of personalization methods to the currently prevailing privacy demands without burdening the application with privacy management tasks.

## References

1. Corbett, A., McLaughlin, M., and Scarpinatto, K. C.: Modeling Student Knowledge: Cognitive Tutors in High School and College. User Modeling and User-Adapted Interaction 10, (2000) 81-108.
2. Strachan, L., Anderson, J., Sneesby, M., and Evans, M.: Minimalist User Modelling in a Complex Commercial Software System. User Modeling and User-Adapted Interaction 10, (2000) 109-146.
3. Linton, F. and Schaefer, H.-P.: Recommender Systems for Learning: Building User and Expert Models through Long-Term Observation of Application Use. User Modeling and User-Adapted Interaction 10, (2000) 181-208.

4. Billsus, D. and Pazzani, M. J.: User Modeling for Adaptive News Access. User Modeling and User-Adapted Interaction 10, (2000) 147-180.
5. Keates, S., Langdon, P., Clarkson, P., and Robinson, P.: User Models and User Physical Capability. User Modeling and User-Adapted Interaction 12, (2002) 139-169
6. Kobsa, A.: Adapting Web Information to Disabled and Elderly Users (invited talk). WebNet-99, Honolulu, HI (1999), http://www.ics.uci.edu/~kobsa/papers/1999-webnet99-kobsa.pdf.
7. Peppers, D. and Rogers, M.: The One to One Future: Building Relationships One Customer at a Time. New York, N.Y.: Currency Doubleday (1993).
8. Peppers, D. and Rogers, M.: Enterprise One to One: Tools for Competing in the Interactive Age. New York, N.Y.: Currency Doubleday (1997).
9. Forrester Research: The Privacy Best Practise. Cambridge, MA Sept. (1999).
10. Hof, R., Green, H., and Himmelstein, L.: Now it's YOUR WEB. Business Week Oct. 5 (1998) 68-75.
11. Personalization & Privacy Survey. Personalization Consortium (2000), http://www.personalization.org/SurveyResults.pdf
12. Thompson, M.: Registered Visitors Are a Portal's Best Friend. The Industry Standard, June 7, 1999, http://www.thestandard.net
13. Brand Conversion. ICONOCAST (1999), http://www.iconocast.com/issue/1999102102.html
14. Recommender Systems in E-Commerce. (2000), http://www.cs.umn.edu/Research/ GroupLens/slides-2.pdf
15. Cooperstein, D., Delhagen, K., Aber, A., and Levin, K.: Making Net Shoppers Loyal. Forrester Research, Cambridge, MA June (1999).
16. Peppers, D., Rogers, M., and Dorf, B.: The One to One Fieldbook. New York, NY: Currency Doubleday (1999).
17. Kobsa, A., Koenemann, J., and Pohl, W.: Personalized Hypermedia Presentation Techniques for Improving Customer Relationships. The Knowledge Engineering Review 16 (2001) 111-155, http://www.ics.uci.edu/~kobsa/papers/2001-KER-kobsa.pdf.
18. Teltzrow, M. and Kobsa, A.: Impacts of User Privacy Preferences on Personalized Systems - a Comparative Study. CHI-2003 Workshop "Designing Personalized User Experiences for eCommerce: Theory, Methods, and Research", Fort Lauderdale, FL (2003), http://www.ics.uci.edu/~kobsa/papers/2003-CHI-teltzrow-kobsa.pdf.
19. Spiekermann, S., Grossklags, J., and Berendt, B.: E-privacy in 2nd Generation E-Commerce: Privacy Preferences versus Actual Behavior. EC'01: Third ACM Conference on Electronic Commerce, Tampa, FL (2001) 38-47, http://doi.acm.org/ 10.1145/501158.501163.
20. A Collection and Systematization of International Privacy Laws, with Special Consideration of Internationally Operating Personalized Websites. (2002), http://www.ics.uci.edu/~kobsa/privacy
21. Teleservices Data Protection Law (Article 3 of the Law on the Legal Requirements for Electronic Business Dealings of 14 Dec. 2001). German Federal Law Gazette 1, 3721 (2001), http://www.iid.de/iukdg/aktuelles/fassung_tdg_eng.pdf
22. Kobsa, A.: Generic User Modeling Systems. User Modeling and User-Adapted Interaction 11 (2001) 49-63 http://www.ics.uci.edu/~kobsa/papers/2001-UMUAI-kobsa.pdf.
23. EU: Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the Protection of Individuals with Regard to the Processing of Personal Data and on the Free Movement of such Data. Official Journal of the European Communities (1995) 31ff, http://158.169.50.95:10080/legal/en/dataprot/directiv/directiv.html.7

24. Brusilivsky, P.: Adaptive and Intelligent Technologies for Web-based Education. KI 4, (2000) 19-25, http://www2.sis.pitt.edu/~peterb/papers/KI-review.html.
25. EU: Directive 2002/58/EC of the European Parliament and of the Council Concerning the Processing of Personal Data and the Protection of Privacy in the Electronic Communications Sector (2002) http://register.consilium.eu.int/pdf/en/02/st03/ 03636en2.pdf.
26. Décret no 99-200 du 17 mars 1999 définissant les catégories de moyens et de prestations de cryptologie dispensées de toute formalité préalable. Le Journal officiel de la République française, (1999) http://www.legifrance.gouv.fr/WAspad/UnTexteDeJorf?numjo=PRMX9903477D.
27. Self-Regulatory Principles for Online Preference Marketing by Network Advisers. Network Advertising Initiative (2000), http://www.networkadvertising.org/images/ NAI_Principles.pdf
28. Schreck, J.: Security and Privacy in User Modeling. Dordrecht, Netherlands: Kluwer Academic Publishers (2003), http://www.security-and-privacy-in-user-modeling.info.
29. Kobsa, A. and Schreck, J.: Privacy through Pseudonymity in User-Adaptive Systems. ACM Transactions on Internet Technology 3 (2003), 149-183 http://www.ics.uci.edu/~kobsa/papers/2003-TOIT-kobsa.pdf
30. GVU's 10th WWW User Survey. Graphics, Visualization and Usability Lab, Georgia Tech (1998), http://www.cc.gatech.edu/gvu/user_surveys/survey-1998-10/
31. Agrawal, R., Kiernan, J., Srikant, R., and Xu, Y.: Hippocratic Databases. 28th International Conference on Very Large Databases, Hong Kong, China (2002), http://www.vldb.org/conf/2002/S05P02.pdf.
32. Enterprise Privacy Architecture: Securing Returns on E-Business. (2002), http://www-1.ibm.com/services/files/epaexecbrief.pdf
33. Karjoth, G., Schunter, M., and Waidner, M.: Privacy-Enabled Services for Enterprises. International Workshop on Trust and Privacy in Digital Business (Trustbus 2002), Aix-en-Provence, France (2002) 483-487.
34. Karjoth, G., Schunter, M., and Waidner, M.: Platform for Enterprise Privacy Practices: Privacy-enabled Management of Customer Data. in 2ndWorkshop on Privacy Enhancing Technologies, *LNCS*. Berlin: Springer-Verlag (2002).
35. Karjoth, G. and Schunter, M.: A Privacy Policy Model for Enterprises. 15th Computer Security Foundations Workshop (CSFW'02), Cape Breton, Nova Scotia, Canada, (2002) 271-281.
36. Fischer-Hübner, S.: IT-Security and Privacy: Design and Use of Privacy-Enhancing Security Mechanisms. LNCS 1958. Heidelberg-Berlin, Germany: Springer (2001).
37. Szyperski, C.: Component Software: Beyond Object-Oriented Programming. Reading, MA: Addison-Wesley (1998).
38. Heineman, G. T. and Councill, W. T.: Component Based Software Engineering: Putting the Pieces Together. Reading, MA: Addison-Wesley (2001).
39. Liu, C.: Redundant Arrays of Independent Components. Irvine, CA: School of Information and Computer Science, University of California (2002).
40. Liu, C. and Richardson, D. J.: Research Directions in RAICs. ACM SIGSOFT Software Engineering Notes 27 (2002).
41. Liu, C. and Richardson, D. J.: The RAIC Architectural Style. School of Information and Computer Science, University of California, Irvine, CA, Working Paper (2002).
42. A P3P Preference Exchange Language 1.0 (APPEL1.0): W3C Working Draft 15 April (2002), http://www.w3.org/TR/P3P-preferences
43. The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. W3C Recommendation 16 April (2002), http://www.w3.org/TR/P3P/

# Privacy in Enterprise Identity Federation – Policies for Liberty Single Signon –

Birgit Pfitzmann*

IBM Zurich Research Lab
bpf@zurich.ibm.com

**Abstract.** Cross-domain identity management is gaining significant interest in industry. A recent example is the Liberty Alliance's specifications for single signon of users across a federation of enterprises. These specifications stress that the federation process is voluntary for the users and that privacy is preserved, e.g., by using pseudonyms. We evaluate the privacy of these specifications in detail. We point out ambiguities and propose a concrete privacy policy together with a few changes to the Liberty processing rules. Our analysis demonstrates that identity-management policies are non-trivial even in a limited context. We also discuss how such low-tech proposals from industry relate to high-tech privacy-enhancing proposals from the research community.

## 1 Introduction

Identity management has many facets. In enterprises, the main emphasis is still on internal consolidation, e.g., on customer-relationship management and on integrating different access channels such as phone and Internet for customers, and Internet and internal systems for employees. In the privacy-research community, the emphasis is on enabling people to manage their identities themselves including free choice of pseudonyms, the transfer of credentials from one pseudonym to another pseudonym of the same person, and appropriate user interfaces. The gap between these facets is wide. Nevertheless, a user-side solution proposed in research can only work if it is taken up on a large scale by enterprises, or if it is compatible on all layers with enterprise-side standards for interacting with users. The latter is quite hard to reach for identity management. (It is easier for mere communication, where some anonymization techniques are transparent to the communication partner.) Furthermore, even though all surveys show that a large majority of the population is concerned about privacy, and about 25% at a considerable price in money or inconvenience, individual users are not good drivers for privacy-enhancing technologies at least given the current ease-of-use and distribution models. This is shown by the results of all companies that tried to commercialize high-end privacy-enhancing technology.

It is therefore essential for the privacy community to also seriously study and try to enhance the privacy achievable by or in interaction with emerging identity-management solutions driven by enterprises. The specifications for single signon across federations of different enterprises recently proposed by the Liberty Alliance

* This paper reflects the view of the author, which is not necessarily shared by IBM.

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 189–204, 2003.

may be such an emerging solution, due to the strong membership in this alliance. Liberty is not an open standardization process, but drafts of the second version, 1.1, are available for public comments.

Detailed privacy studies are also important for the enterprises involved in emerging standards because a lack of user trust is a major inhibiting factor for electronic commerce. In other words, a large group of users who are not sufficiently motivated to buy user-side privacy technology are nevertheless sufficiently worried not to use a lot of enterprise-side technology. Specifically for single signon and federated identities, market studies in the wake of Microsoft's Passport product corroborate this clearly. Indeed, the Liberty specifications stress that federating, i.e., choosing single signon between two enterprises, is voluntary for the users and that privacy is preserved, e.g., by using pseudonyms. The minimum goal of such a standard with respect to privacy should be clarity: If users believe in stronger privacy than enterprises do, the users will feel cheated and may start litigation. If enterprises believe in stronger privacy than users do, users will be more reluctant to use the protocols than they need to be.

When looking at Liberty from a privacy perspective, one should be aware that its focus is a business-to-business scenario with small federations with close trust relationships, called circles of trust. (The current protocols do not even scale to large and multiple federations due to assumptions about initial key distribution and specific message formats.) This is different from hosting general-purpose end-user wallets, which is the focus of Microsoft's Passport and of high-tech privacy-enhanced identity management. Liberty's example use case is a federation of airlines and rental-car companies, and a user who already has accounts with two federation members and wants to link them. For instance, bonus points might then accumulate. One could implement a bonus point system nicely with cryptographic credentials, but real airlines and rental-car companies require the user's name and address and relatively strong identification and will not be easily persuaded out of this. Thus most users have to trust these organizations anyway not to exchange undesired information about them, i.e., they have to trust these organizations' explicit or unwritten privacy policies. Of course, there are also business-to-business scenarios where a user wants to be unlinkable even within a small federation or in different interactions with one enterprise.

Studying the privacy provided by a single-signon protocol like Liberty's has two goals: First, make sure that the privacy policies and implications are clearly specified. This is a completely technical goal. Secondly, discuss whether these policies, including the given user options, are suitable for the stated purposes. Indeed we will point out several major and minor privacy-related ambiguities in the Liberty specification. We will propose fixes and an overall policy for the Liberty specifications. A third potential goal is out of scope of our paper: to compare such policies with exact privacy regulations for different countries and sectors. However, we hope that our technical work can serve as a basis for such legal studies.

### Overview of This Paper

In Section 2, we survey related literature, and in Section 3 we give an overview of the protocol we analyze. In Section 4, we summarize the ambiguities. This is an addition to the introduction, using terminology explained in Section 3. In Section 5, we approach the policy question for Liberty systematically by studying the data that are

released in the protocols given certain user choices, and whether this is fully specified or not. In Section 6, we discuss what privacy policies best fit the Liberty protocols. Section 7 gives an outlook and Section 8 a summary.

## 2 Related Literature

The Liberty specifications are one of several recent specifications of web single signon across different enterprises for users that have nothing but a browser. Accommodating this "zero-footprint" case, at least among others, is currently considered essential for market acceptance. Three parts of the six-part Liberty specification are relevant for us [Lib02, Lib02a, Lib02b][1]. Such browser-based protocols were initiated by Microsoft's Passport product [Mic01], which led to many discussions about privacy and points of control. The only technical contributions mainly concerned operational security [KR00, Sle01]. For a similar product from another company, see [IBM02] Ch. 10.2. The only open standardization initiative is OASIS's SAML [SAM02]. Both Liberty and the Internet2-project Shibboleth [Shi02] are built upon SAML.

We gave an overview of privacy requirements and design consequences for browser-based protocols in [PW02], together with a sketch of a protocol BBAE achieving optimal privacy. (BBAE is available in more detail in [PW02a].) That overview concentrates on general-purpose attribute-exchange protocols, and on design consequences from privacy on message flows and formats. It does not propose specific policies for the choice of names and attributes. The current paper does this for one specific protocol. We chose Liberty for this analysis because it is single-signon only, not very extensible, and even defines some user-interface aspects, i.e., it is really one protocol only, in contrast to the high flexibility of SAML or BBAE. Moreover, it is surprising how complicated a policy becomes even for pure single signon. We guess that the Liberty Alliance postponed attribute-exchange protocols in order to avoid policy issues. The analysis shows that this separation is not possible.

The high end of privacy-enabled identity management is exemplified by the idemix prototype of an anonymous credential system [CL01, CV02]. In particular, this is the first system with efficient multi-show credentials, comprehensive choices as to anonymity revocability, and an all-or-nothing transfer property. Anonymous credentials were first proposed in [Cha85]. A high-level vision of an overall system built around these ideas and other, complementary privacy techniques like anonymous communication, is given in [CPH+02]. Anonymous credentials are the only known identity-management solution for cases where a user wants to use unlinkable pseudonyms with different organizations and nevertheless transfer certified attributes between these organizations. However, in current electronic commerce, not many attributes are certified; typically users just fill in forms, and even Microsoft Passport did not certify anything until quite recently, and now only control of email addresses. Hence one can strive for using simple pseudonyms where no certification is needed, simple certificates where no anonymity is possible for other reasons (which may be non-technical and thus changeable), and anonymous credentials in the remaining case.

[1] Liberty has made drafts of V1.1 available (http://www.projectliberty.org/specs/v1_1draft/index.html). All our citations remain to the stable version 1.0, but we verified that no essential changes were made or announced to the parts that we discuss (as of Nov. 29).

For a discussion of more remotely related techniques like form fillers and PKIs we refer to the appendix of [PW02]. The pseudonym-choice policies of Liberty resemble [GGK+99]. However, there the single-signon provider acts as a browser proxy. While browser proxies are a good choice for user-side identity management, they would be very privacy-unfriendly for enterprise identity management as in Liberty: If one enterprise acts as a user proxy with respect to another enterprise, it sees the user's entire communication with the second enterprise. In contrast, in all browser-based protocols mentioned above, the single-signon provider only takes part as a server specifically during single signon. Further, this allows the use of multiple single-signon providers per user, e.g., a bank for financial information, a doctor for medical information, and a user-side wallet for personal information. (However, Liberty itself, in contrast to BBAE, does not allow user-side wallets, corresponding to its scenario of small enterprise federations only.)

## 3 Liberty Single Signon and Federation

We first introduce browser-based single-signon protocols in general, and then special aspects of the Liberty protocol.

### 3.1 Browser-Based Single Signon

The overall structure of all current browser-based single-signon and attribute-exchange protocols is shown in Figure 1.

A user is initially browsing at a service provider. When the user wants to log in (or to send attributes in more general protocols), the service provider redirects the browser to the user's identity provider. The user logs in there, typically with a fixed user ID and password. The browser and identity provider may also reuse a secure session from another recent login. The identity provider then redirects the browser back to the service provider with some ticket. If the information to be transferred is short, it can be completely included in this ticket. Most protocols also provide a back channel for transferring longer information; the ticket then contains a handle to that information so that the service provider can associate a returning browser with the appropriate back-channel information.

### 3.2 Special Aspects of Liberty Federation

The overall Liberty scenario is described in [Lib02]. A user starts participating by consenting to "federation" of "introductions" at an identity provider. Such a phase is normally called registration; only in Liberty it is assumed that the user already has an account at the identity provider, so that no new data are exchanged. Later, when the user browses at a service provider in the same federation, the service provider notices that the user has an identity provider in this federation, and asks the user whether he wants to federate or link these two specific accounts ("identities"). The only subprotocol specified for how the service provider notices this is by a cookie in a common federation domain; we assume in the sequel that this subprotocol is used. Then the

redirections as in Figure 1 happen for the first time, and the identity provider and service provider exchange a pseudonym by which they will refer to this user. Later signon happens under this pseudonym, again according to Figure 1. The message formats are described in [Lib02a] and the protocols ("profiles") in [Lib02b].

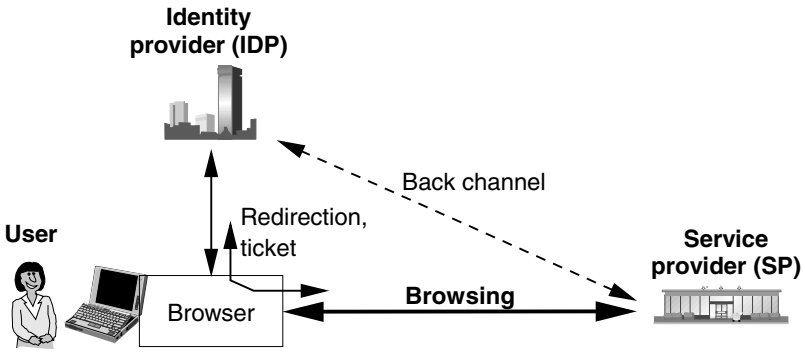



**Fig. 1.** Scenario of browser-based single signon.

### 3.2 Special Aspects of Liberty Federation

The overall Liberty scenario is described in [Lib02]. A user starts participating by consenting to "federation" of "introductions" at an identity provider. Such a phase is normally called registration; only in Liberty it is assumed that the user already has an account at the identity provider, so that no new data are exchanged. Later, when the user browses at a service provider in the same federation, the service provider notices that the user has an identity provider in this federation, and asks the user whether he wants to federate or link these two specific accounts ("identities"). The only subprotocol specified for how the service provider notices this is by a cookie in a common federation domain; we assume in the sequel that this subprotocol is used. Then the redirections as in Figure 1 happen for the first time, and the identity provider and service provider exchange a pseudonym by which they will refer to this user. Later signon happens under this pseudonym, again according to Figure 1. The message formats are described in [Lib02a] and the protocols ("profiles") in [Lib02b].

## 4 Overview of Ambiguities

Before starting the somewhat tedious data analysis, we summarize the main questions that will remain open in Section 5, i.e., for which we will not only extract an existing implicit Liberty policy.

- If a user consents to the federation of two identities that she is using with two organizations, does she consent only to single signon between them, or to arbitrary background information sharing, or does this depend on policies?
- Can a service provider federate, i.e., link accounts, even without user consent?
- How can a service provider restrict single signon to situations after federation?

# 5 Data Released in Liberty Protocols

We now approach the privacy question systematically by looking at the types of consent in Liberty V1.0 and the technical consequences, in order to identify which data releases the Liberty policies must cover as a minimum.

## 5.1 Liberty Consents

A user in Liberty has two choices, corresponding to opt-ins to certain data releases:
1. When logged in at an identity provider *IDP*, the user can allow to federate the current identity in principle. We call this "introduction consent".
2. When interacting with a service provider *SP*, the user can allow to federate her current identity with that at a specific identity provider *IDP* of the same federation. We call this "federation consent". The use cases suggest that the user must be logged in at *SP*, and thus a priori known at *SP* (in particular [Lib02], p.10), but technically nothing is based on this.

## 5.2 Liberty Data Releases

The following data releases happen in the Liberty protocols.

### 5.2.1 Directly after Introduction Consent

Upon introduction consent, *IDP* sets a cookie on the user's browser in a federation domain. Recall that we assume that the only specified protocol for this phase is used; it is the "Identity Provider Introduction" from [Lib02b], Section 3.6. This tells all other members of this domain that the user has identity provider *IDP*.

It is explicitly left open whether the cookie is persistent or session-based, i.e., whether it only divulges the name *IDP* or the fact that the user is currently logged in at *IDP*.

If new members join the federation, they automatically also receive introductions by this technique.

### 5.2.2 Federation

At any time after introduction consent, *IDP* is willing to accept federation requests from service providers in its federation. A federation request is a single signon request with an element `<Federate>=true` [Lib02a].

The first problem is whether introduction consent also allows the identity provider to federate with a specific service provider. The Policy/Security Note after Figure 4 in [Lib02] strongly suggests that it does not: "In Figure 4 the user is not consenting to federating his identity with any service providers. Soliciting consent to identity federation is a separate step, as illustrated in Figure 5." However, technically it does: Assume user *U* wants to enable federation at an identity provider *IDP* of a federation *F*, but not with federation member *SP* because *U* does not fully trust *SP*. However, the following consent for federating with *SP* is only given to *SP*, and *IDP* simply believes it when obtaining the federation request. Hence if *SP* is indeed untrustworthy, it can

get the federation without *U*'s consent. By using the element `<IsPassive>` in the request, *SP* can even ensure that *IDP* does not contact *U* during this request, so that *U* cannot notice this and complain.

Upon a federation request, *IDP* generates a new pseudonym $id_{U,SP}$ for user *U* in interaction with *SP*, and is from then on willing to always authenticate *U* to *SP* under $id_{U,SP}$[2].The rules for pseudonym generation are missing in the processing rules for the request, Section 3.2.3 of [Lib02a], but are defined at the beginning of Section 3.3: "At the time of federation, the identity provider generates an opaque handle that serves as the name identifier the service provider and the identity provider use in referring to the Principal when communicating with each other."

### 5.2.3 Starting Single Signon

After federation consent, *SP* makes a federation request to *IDP*, and can then ask *IDP* to authenticate *U* at any future time. The text is quite specific that single signon should only be used after federation ([Lib02], Sections 2.2 and 5.4.2). However, it is not specified how *SP*, at the moment where it desires single signon, knows whether it has federated for this user: Even if *SP* has an account for *U* and noted the federation there, at this moment *U* has not been authenticated. A possibility is that *SP* has set a persistent cookie on *U*'s browser and desires the single signon only for better security. However, the cookie may be absent because *U* switched cookies off or uses multiple browsers.

- Now either *SP* might simply try single signon, implying certain data releases (see Section 5.2.4) even if *U* did not consent to federation.
- Or *SP* asks for user consent now; we call this "single-signon consent".

### 5.2.4 Data in Single Signon

In each single signon *SP* tells *IDP* that *U* is currently browsing there, and gets *U* authenticated under $id_{U,SP}$. More precisely, first *IDP* obtains the information that the user whom *IDP* knows under a local name $id_{U,IDP}$ is currently browsing at *SP*, while *SP* itself does not know this. This happens transparently if *U* is authenticated at *IDP* at that time; otherwise *U* gives some additional implicit consent by authenticating to *IDP*. Then *IDP* tells *SP* that this is the user known as $id_{U,SP}$ at *SP*.

### 5.2.5 Attribute Exchange

No exchange of further attributes of *U*, in particular of names or addresses, happens directly by the Liberty protocols. However, the existence of a common pseudonym enables the identity provider and the service provider to exchange data about the user by other protocols. The main question, as anticipated in Section 4, is whether federation consent for two organizations implies

a) consent only to single signon between the two organizations,
b) or to arbitrary background information sharing between them,
c) or whether this, and the extent of the sharing, depends on policies specific to the federation or the current two organizations.

---

[2] *SP* may ask to have this pseudonym replaced by another one, using the name registration protocol, but this makes no difference for privacy, only that *SP* may need one name less internally.

This must be stated clearly, and in Case c), a clear user interface is needed for looking up these policies prior to consenting. To see that this is indeed unclear, compare the following information:

- From the accompanying press release (http://www.projectliberty.org/press/releases/2002-07-15-1.html): "The Liberty version 1.0 specifications do not involve the exchange of personal information. Instead, they involve a format for exchanging authentication information between companies so the identity of the user is safe, and specific details about the customer's identity are not shared." This sounds like Case a), and most users will understand it like this and expect background information-exchange to be forbidden. However, one can take the position that also in Case b) and c), the *specifications* do not involve what happens in the background.
- The discussion that providers cannot skip over each other in chains of linked identities in [Lib02], Section 5.4.1, also sounds like Case a), because it suggests that no other attributes like names are exchanged that would be the same throughout the chain.
- The notion of "account linking" in [Lib02], Section 2, and the general network-identity vision in Section 1.2, sound like Case b).
- The unspecified background web services in [Lib02], Figure 11, look like Case b), but instead they may only be the federated logout service.
- After Figure 17 in [Lib02]:"The semantics of such a federated relationship between identity providers are not dictated by the underlying Liberty protocols. These semantics will need to be addressed by the agreements between the identity providers and the capabilities of the deployed Liberty-enabled implementations." This sounds like Case c), although only for the special case of two identity providers.
- The example screen asking the user for federation consent, Figure 5 in [Lib02], contains no link for help or for looking up the federation rules. This looks as if it cannot be Case c).

### 5.3 Withdrawing Consent

No protocol is specified for undoing introduction consent, but it can easily be added by asking the identity provider to delete the cookie. We call this "revoking introduction consent". As all other interactions can be terminated by the user, we assume that this is also intended in Liberty. Users may also want to revoke introduction consent for reasons other than privacy, in particular for changing their identity provider.

Federation consent can be undone at either *IDP* or *SP* by "federation termination", also called "defederation" (Section 5.4.1.2 in [Lib02] and Section 3.4 in [Lib02b]). Either party has to notify the other, and from then on, *SP* should not send authentication requests to *IDP*, nor *IDP* answer them from *SP*. If *SP* and *IDP* later federate again, *IDP* generates a new pseudonym as described above. This implies that by federation termination at *IDP*, user *U* can reliably cut the link between its prior interactions with *SP* and future ones (while interacting in two roles with *SP* in parallel is not possible). Of course, if federation occurs both times from an existing account with *SP*, then *SP* can link all interactions with *U* anyway.

# 6 Policy Proposals

Now we propose concrete policy rules, structured according to the different data categories that we have seen. We sometimes present several options, but we propose to fix one clear policy for usage with the Liberty V1.0 specifications or similar specifications, leaving a larger choice of policies to a future version that also offers a larger choice of techniques. In other words, simplicity is the main advantage of having only single signon, fixed federations, and fixed roles, and this should be reflected by a clear and simple policy.

## 6.1 Overview Table

Table 1 gives a detailed overview of the data types that we saw and some that might occur in future extensions, together with the main policy options as discussed already and in the rest of Section 6. A short summary of the final recommendation will be given in Section 8.

## 6.2 Introduction Data

The data that introduce an identity provider are not very sensitive, in particular as long as they do not contain any details such as the responsibilities of a particular identity provider with respect to a user. Releasing them improves user convenience, which seems the main motivation for a pure single-signon protocol. Hence we propose a lax privacy rule:

**Rule$_{\text{intro}}$:** If user *U* gives introduction consent, *IDP* may tell arbitrary recipients where *U* is browsing that this user has identity provider *IDP*. This holds until future opt-out by revoking introduction consent; an easy interface for this must be provided.

Note that this release is not linked to any name or pseudonym of *U*, just to "the current browser user". Furthermore, we have allowed release to all future federation members as needed for the Liberty introduction protocol, and even to outsiders, so that the common-domain cookie does not need special protection. (Actually, the Liberty recommendations do provide this stronger protection.) We decided that the introduction data should be the name *IDP*, not the current login status, mainly because it is more convenient for the user if single signon also starts if the user is not logged in at *IDP* yet.

## 6.3 Authentication Data

Authentication data are much more critical than introduction data. We recommend the following rule for identity providers:

**Rule$_{\text{IDP,auth}}$:** Introduction consent is for introduction only, without any effect on authentication. User *U* can separately give federation consent at *IDP* for every specific service provider *SP* to allow federation with *SP*. Then *IDP* may authenticate *U* under a fresh, but then fixed role pseudonym $id_{U,SP}$ to *SP* whenever *SP* asks, until future opt-out by federation termination.

**Table 1.** Overview of data categories and recommended policy rules for their release.

| Data Category | Detailed data | Consent needed at | Exists in Liberty SSO? | Realization (Liberty SSO or extensions) | Recommendation for Liberty SSO policy | Recommended side effects of rule |
|---|---|---|---|---|---|---|
| General | | | | | | Later opt-out. Minimum dispute resolution and assurance. |
| Introduction | *U*'s identity provider | *IDP* | Yes, or 3 | Cookie from *IDP* (or header, script etc.) if persistent | Do after introduction consent | Access clear by cookie. Retention limit. |
| | Responsibility of *IDP* (e.g., financial or work) | *IDP* | No | Cookie etc. as above | n/a (= not applicable) | n/a |
| | Login status | *IDP* | Yes, or 1 | Cookie etc. if per session | Don't | n/a (else retention limit) |
| Traffic | Name of *SP* and fact that *U* is now browsing there | *SP*. For transparent use via IDP? | Yes | Needed for redirect back and man-in-the-middle security | Do after federation consent or single-signon consent at *SP* | Implicit access to *SP*-names, retention limit. (If at *IDP*, audit and harsh punishment for abuse) |
| | Anything else about *U*'s current actions at *SP* | *SP* | Maybe | Exact accessed URL; element `<RelayState>` | Don't | n/a (else retention limit) |
| Names | Fixed role name per service provider | *IDP*. For transparent use via *SP*? | Yes | Name in authentication token, or name or attribute in attribute token | Do after federation consent at *IDP* | Retention limit. (If at *SP*, audit and punishment for abuse.) |
| | Name with a-priori meaning | *IDP* | No | As above | n/a | n/a |
| | Freely chosen role | *IDP* | No | As above | n/a | n/a |
| Other user attributes | Arbitrary | *IDP* (and *SP* if bidirectional exchange) | Maybe | Attribute tokens, or background exchange enabled by linking | Consent to policy with federation consent at *IDP*. Only *IDP*. | Require a seal for policy. Details in policy. |
| | What attributes *SP* wants | *SP* | Maybe | Attribute queries | Consent to policy with federation consent at *SP* | Require a seal for policy. Details in policy. |

This rule implies a small change to Liberty's processing rules (Section 3.2.3 of [Lib02a]): *IDP* itself must ask for a user OK when federating with a new service provider *SP*. This is fourth type of consent after the original two from Liberty (Section 5.1) and the potential single-signon consent from Section 5.2.3. In future versions with flexible protocols and privacy policies, a user *U* can also pre-authorize this for arbitrary sets of service providers.

The following rule is closer to the current Liberty specifications, but only our second choice:

**$\text{Rule}_{\text{IDP,auth,2nd}}$:** If user *U* gives introduction consent at *IDP*, then *IDP* may authenticate *U* under a fresh, but then fixed role pseudonym $id_{U,SP}$ to any service provider *SP* that was a member of *IDP*'s federation *F* at the time of *U*'s choice. The list of members of *F* must be easily retrievable when *U* makes the choice. This holds until future opt-out by federation termination.

The restriction to federation members at the time of consent in this rule is necessary to keep *U* in control. Otherwise the risk is large that some federation will grow to include almost every company in the world, and suddenly *U* will be authenticated under fixed pseudonyms with service providers where she never intended that. Disadvantages of this rule over our recommendation are:

- Identity providers must store for which federation members *U* gave consent, and will probably want a user interface for consent for new members, so that all technical additions needed for $\text{Rule}_{\text{IDP,auth}}$ are also needed here.
- In larger federations, almost no user will want federation with all service providers, both for privacy and for convenience (this type of federation does not allow multiple roles).
- Audit and minimum punishments are otherwise needed to make $\text{Rule}_{\text{SP,auth}}$ credible, i.e., to deter dishonest service providers from federating with *IDP* against the user's wish.
- It works less well together with the rule we propose for attributes (Section 6.5).

For the service provider, introduction consent has no effect (for both versions of the rule for the identity provider.) In particular, *SP* should not collect introduction data or contact *IDP* for users that do not choose to federate at *SP*.

**$\text{Rule}_{\text{SP,auth}}$:** If user *U* gives federation consent for an identity provider *IDP* at *SP*, then *SP* may record this choice (e.g., by setting cookies on the user's browser) and the pseudonym $id_{U,SP}$ of this user, and may link different interactions with this user by this pseudonym. If the choice was made from an existing account, it may also link these interactions to the existing account.

### 6.4 Traffic Data

Traffic data arise at the identity providers because service providers notify the identity provider that the user is browsing there. These data arise implicitly in the protocol and are not needed for the applications that a user expects. Hence we propose strict privacy rules for them.

In particular we require that a service provider who is not sure about federation consent asks the user for single-signon consent, i.e., we opt for the second solution to the problem from Section 5.2.3. Interaction with *U* is typically needed anyway in this

situation because the *IDP*-introducer cookie is also absent; such interaction is described in [Lib02], Section 5.4.3.5. An advantage of this solution is that the alternative would need strict audit whether some identity providers introduce themselves for users that never gave introduction consent there, in order to collect visited-sites trails from service providers in this situation.

**Rule$_{IDP,traffic}$:** (Given introduction consent.) An identity provider *IDP* must not mine traffic data or use them for any other purpose than single signon. He must not forward them to any other party. Exceptions may only be given by law (e.g., storage requirements for law enforcement) and for authentication classes where dispute resolution is offered (i.e., where a service provider and a user may need records from *IDP* about an authentication that the user denies).

**Rule$_{SP,traffic}$:** (Given federation consent or single-signon consent.) A service provider *SP* must only provide fixed data about itself to the identity provider in single signon and federation, not user-dependent data.

Consequently, *SP* should not put unencrypted user data in the element `<RelayState>` of a single-signon request, in particular not the exact URL that the user wanted to access. The contradicting recommendation in Section 3.2.1 of [Lib02b] should be modified (in Step 3, recommending to use the element `LRURL=<return URL>` from Step 1.) Random values and data encrypted with a key known only to *SP* are permitted.

### 6.5 User Attributes

Now we come to the question of user attributes. As they do not occur explicitly in the Liberty V1.0 specifications, we are technically free to decide about them. We will recommend a certain version of Case c) from Section 5.2.5, i.e., identity-provider-specific policies.

From a privacy perspective it is tempting to recommend Case a) instead, i.e., essentially no attribute sharing; hence we discuss this first. The rule could look as follows:

**Rule$_{attributes,(a)}$:** No consent implies any permissions beyond the policy rules from Sections 6.2 to 6.4. In particular, an identity provider or service provider must not share any data about a user they both know as $id_{U,SP}$ using this pseudonym, beyond what is allowed in these rules, nor must they use the common domain for any cookies beyond the specified introduction cookies.

The term “using this pseudonym” already weakens the rule by allowing other forms of sharing. This is unavoidable in particular in closed federations, because some federation members will already be sharing information about these users, e.g., employers and travel agents about traveling employees.

While this rule is reasonable in itself, we cannot imagine that typical federations offer single signon with so little benefit to themselves. The other extreme, an implicit permission to share arbitrary data (Case b), is out of the question because it contradicts all privacy principles, such as first formulated in [Wes67]. We therefore assume that the Liberty Alliance meant some form of Case c). This, however, requires a reference to a privacy policy. It seems impossible to propose just one policy (even with open parts for user choices) for all federations. Hence we recommend the following rules:

**$Rule_{IDP,attributes,(c)}$:** If user *U* consents at *IDP* to federate with a service provider *SP*, then *IDP* may use the generated pseudonym $id_{U,SP}$ to provide information about *U* to *SP*, provided *U* also consented to a privacy policy that allows this. The policy must be explicitly referred to and easy to look up in detail before the consent, and withholding consent must be easy. The policy should at least have a seal from a well-known organization. The permission holds until future opt-out by defederation. To what extent sent attributes may survive defederation must be clarified in the policy, as well as to what extent *IDP* may store a history of *SP*'s requests.

**$Rule_{SP,attributes,(c)}$:** If user *U* consents at *SP* to federate with an identity provider *IDP*, then *SP* may use the provided pseudonym $id_{U,SP}$ to ask *IDP* for attributes about *U* and to use the obtained attributes, provided *U* also consented to a privacy policy that allows this. The policy must be explicitly referred to and easy to look up in detail before the consent, and withholding consent must be easy. The policy should at least have a seal from a well-known organization. The permission holds until future opt-out by defederation. To what extent received attributes survive defederation must be clarified in the policy.

**$Rule_{cookies}$:** The common domain is not to be used for cookies except the specified introduction cookies.

The following points should be noted about these rules:

- The rules are asymmetric, i.e., they assume that attributes are only transferred from *IDP* to *SP*. This is more user-friendly given that a distinction between identity providers and service providers is made. For bidirectional exchange, the organization *SP* should also act as an identity provider towards the organization *IDP*, i.e., get separate user consent for sharing its own data.
- As in $Rule_{attributes,(a)}$, the organizations may still have separate legacy processes for sharing data in ways that may no longer even be known and that therefore have no privacy policy. However, all processes that use $id_{U,SP}$ are necessarily new and known. Therefore it is reasonable to set up a privacy policy when setting up these processes.

### 6.6 Further Data?

We believe that we have covered all data occurring in the context of the Liberty V1.0 specifications, except for traffic analysis possible in networks. This is not made significantly easier by these protocols if all data are sent over secure channels, and thus we do not consider it further. Extensions of the protocols to larger or more dynamic federations could include key distribution centers or other central directories; if this is done in a way that enables the centers to collect usage trails, the policies put up for consent must govern these data.

### 6.7 Further Policy Aspects

So far, we have only considered rules for the data releases from one party to another. Privacy policies also govern other aspects, in particular conflict-resolution procedures, user-access rights, notification, and retention periods. In addition to these func-

tional elements, there can be assurance elements such as promises of regular audit, enterprise-internal need-to-know policies, or security evaluations. While we also propose that Liberty fixes as much of this as possible to retain the simplicity of V1.0, we only sketch our recommendations:

- **Termination:** For all consent, the possibility for a later opt-out is already provided.
- **Dispute resolution:** A minimum standard for dispute resolution should be set. We recommend at least a contact address at each IDP and a fixed address per federation as a second resort. Further, while law suits are then hopefully not needed, it seems clear that breach of privacy promises can be a basis for litigation independent of whether this is offered as an explicit dispute resolution type in the policy.
- **Notification:** Under the recommended policy, no specific user notification is needed, because all releases are governed by policies that were consented to at the time of the first such release. Notification about certain attribute releases can be promised indirectly in the attribute-release policy.
- **Access rights for the user:** Under the recommended policy, no specific user access rights are needed[3]. Access rights for certain attribute releases can be promised indirectly in the attribute-release policy. Access to the names of service providers that were federated with an identity provider *IDP* is useful, but must be given anyway in the user interface for defederation at *IDP*.
- **Retention periods:** Each identity and service provider should state retention periods for all data covered by these policies. In addition, a general upper bound on the retention of traffic data seems useful, e.g., one month unless local law or the authentication class require a longer period.
- **Assurance:** Certain minimum assurance standards should be fixed at least for identity providers. Attribute policies at identity providers can additionally require assurance for service providers that receive certain attributes.

## 7 Outlook

We already sometimes mentioned future extensions to larger and more dynamic federations and to protocols with integrated attribute exchange. The recommended rules should scale well to those cases. We foresee the following most important differences:

- Attribute-exchange protocols can deal better with incomplete policies. This is important for scenarios where an identity provider mainly serves as a trusted user agent following a user-chosen policy, because most people are neither willing nor able to initially set the entire policy of what information they want to share with whom. (In contrast, in some closed federations like supply chains, the users are mainly agents of the federation partners. Then the policy can be fixed by the identity providers.) The attribute-exchange protocol can then contain a real-time release of the attributes.

[3] In contrast, policies where a first party believes a second party that a user gave consent at the second party need access rights to enable misuse detection. This would hold for the second-choice rule $\text{Rule}_{\text{IDP,auth,2nd}}$ and if $\text{Rule}_{\text{SP,traffic}}$ did not have the condition of federation or single-signon consent.

- Attribute-exchange protocols also enable the provision of demographic or preference data about an anonymous user for whom not even a long-term pseudonym is provided.
- Authentication information will become a special case of attributes, because users may have more than one pseudonym with one service provider, or the same pseudonym with several service providers. In other words, a Liberty V1.0-style pseudonym is just one type of name.
- In the general case, there is no need for a special semantics of "federate" any more, i.e., the user choices do not need to be bundled in the same way. (This may remain one option.)
- In a general e-commerce scenario, users should also be allowed to be their own identity providers, in particular for voluntary attributes like preferences, i.e., to have user-side wallets. Then the names, addresses, and keys of these wallets are personal information and must be covered by the policies. How this reflects into the protocol design can be found in [PW02].

**Table 2.** Summary of recommended policy rules for Liberty V1.0.

| Data category | Detailed data | Consent needed at | Exists in Liberty SSO V1.0? | Recommendation for Liberty SSO policy | Recommended side effects of rule |
|---|---|---|---|---|---|
| General | | | | | Later termination; minimum dispute resolution and protection standards |
| Introduction | *U*'s identity provider | *IDP* | Yes | Do after introduction consent | Retention limit |
| | Login status | *IDP* | Maybe | Don't | n/a |
| Traffic | Name of *SP* and fact that *U* is now browsing there | *SP* | Yes | Do after federation or single-signon consent at *SP* | Retentiosn limit |
| | Anything else about *U*'s current actions at *SP* | *SP* | Maybe | Don't | n/a |
| Names | Fixed role name per *SP* | *IDP* | Yes | Do after federation consent at *IDP* | Retention limit |
| Other user attributes | Arbitrary | *IDP* | Maybe | Consent to policy with federation consent at *IDP* | Require a seal for policy. Details in policy. |
| | What attributes *SP* wants | *SP* | Maybe | Consent to policy with federation consent at *SP* | Policy mainly covers *SP*'s usage of received attributes |

## 8 Summary

We have analyzed the privacy effects of a web single-signon and identity-federation protocol, specifically the Liberty V1.0 specifications. Although single signon seems quite a fixed notion in contrast to more general attribute exchange, there were a num-

ber of privacy ambiguities, and we discussed options for resolving them. We proposed precise recommended policy rules and some alternatives; the recommended policy is summarized in Table 2. We described small changes to Liberty's processing rules needed to support this policy, in particular two new types of consent.

## Acknowledgements

Thanks to Matthias Schunter and Michael Waidner for interesting discussions about these policy recommendations.

## References

Cha85 David Chaum: Security without Identification: Transaction Systems to make Big Brother Obsolete; Communications of the ACM 28/10 (1985) 1030-1044

CL01 Jan Camenisch, Anna Lysyanskaya: An efficient system for non-transferable anonymous credentials with optional anonymity revocation; Eurocrypt 2001, LNCS 2045, Springer-Verlag, Berlin, 93-117

CV02 Jan Camenisch, Els Van Herreweghen: Design and Implementation of the Idemix Anonymous Credential System; 9th ACM Conference on Computer and Communications Security (CCS), 21-30

CPH+02 Sebastian Clauß, Andreas Pfitzmann, Marit Hansen, Els Van Herreweghen: Privacy-Enhancing Identity Management; The IPTS Report (67) 2002, http://www.jrc.es/pages/iptsreport/vol67/english/IPT2E676.html

IBM02 IBM: Enterprise Security Architecture using IBM Tivoli Security Solutions; April 2002, http://www.redbooks.ibm.com/abstracts/sg246014.html

KR00 David P. Kormann, Aviel D. Rubin: Risks of the Passport Single Signon Protocol; Computer Networks 33 (2001) 51-58

Lib02 Liberty Alliance Project: Liberty Architecture Overview, Version 1.0, 11 July 2002

Lib02a Liberty Alliance Project: Liberty Protocols and Schemas Specification, Version 1.0, 11 July 2002

Lib02b Liberty Alliance Project: Liberty Bindings and Profiles Specification, Version 1.0, 11 July 2002

Mic01 Microsoft Corporation: Various .NET Passport documentation (started 1999), in particular Technical Overview, Sept. 2001, and SDK 2.1 Documentation; http://www.passport.com and http://msdn.microsoft.com/downloads

PW02 Birgit Pfitzmann, Michael Waidner: Privacy in Browser-Based Attribute Exchange; ACM Workshop on Privacy in the Electronic Society, Washington, Nov. 2002, post-conference proceedings to be published by ACM

PW02a Birgit Pfitzmann, Michael Waidner: BBAE – A General Protocol for Browser-based Attribute Exchange; IBM Research Report RZ 3455 (# 93800), Sept 2002, http://www.zurich.ibm.com/security/publications/2002.html

SAM02 OASIS Security Assertion Markup Language (SAML); Committee specification 01, May 2002 (started Jan. 2001), http://www.oasis-open.org/committees/security/docs

Shi02 Shibboleth-Architecture Draft v05; May 2002, http://middleware.internet2.edu/shibboleth/docs/draft-internet2-shibboleth-arch-v05.pdf

Sle01 Marc Slemko: Microsoft Passport to Trouble; Rev. 1.18, Nov. 2001 http://alive.znep.com/~marcs/passport/

Wes67 Alan F. Westin: Privacy and Freedom; Atheneum, New York NY, 1967

# From P3P to Data Licenses

Shi-Cho Cha and Yuh-Jzer Joung

National Taiwan University, Taipei, Taiwan
csc@mba.ntu.edu.tw, joung@ccms.ntu.edu.tw

**Abstract.** P3P provides a standard means for Web sites to disclose their privacy policies when they need users' personal data for processing. A user can then decide whether or not to provide personal data to the sites based on the disclosed policies. The decision process can also be made automatic through an agent or browser via the privacy preferences set by the user. As can be seen, however, this mechanism cannot guarantee that Web sites *do* act according to their policies once they have obtained user's personal data. In light of this, we proposed a new technical and legal approach, called *Online Personal Data Licensing* (OPDL). The idea is that the use of a person's data must be authorized by the person through the issue of data licenses. Licenses can then be checked to prevent personal data from being misused. This paper focuses on the implementation of OPDL. As P3P provides a standard format for expressing privacy practices about personal data, we use it here to implement data licenses.

## 1 Introduction

Accompanied with the hope of industry and individuals, the first formal specifications of Platform for Privacy Preferences (P3P) was proposed by the World Wide Web Consortium (W3C) in April 2002 [1]. P3P provides a standard means for Web sites to disclose their privacy policies when they need users' personal data for processing. A user can then decide whether or not to provide personal data to the sites based on the disclosed policies. The decision process can also be made automatic through an agent or browser via the privacy preferences set by the user. As can be seen, however, this mechanism cannot guarantee that Web sites *do* act according to their policies once they have obtained user's personal data [2,3].

This critique applies to other non-P3P compatible Web sites as well. In fact, current Web sites are usually required to disclose their privacy policies online, regardless of whether or not they choose P3P to state their policies. If a user continually uses a site's services or even registers as a member of the site, then this will be considered as an indication that he is aware of the site's policies, and has given his consent to the site for allowing the site to collect and use his personal data. This type of 'passive consent' raises lots of privacy disputes as (1) most people have neglected the disclosed privacy policies, and (2) disclosing privacy policies does not guarantee faithful execution of them.

R. Dingledine (Ed.): PET 2003, LNCS 2760, pp. 205–221, 2003.

In light of these problems, we proposed a new technical and legal approach called *Online Personal Data Licensing (OPDL)* in [4]. The idea is that the use of a person's data must be authorized by the person. By allowing people to issue licenses for the use of their data, they can control the way their data are to be used, and be more alert to the exposure of their personal data. Moreover, licenses can be used in the following ways to enforce individuals' consent about privacy practices:

- Licenses can be used to prevent personal data from being used without individuals' consent. To do this, we can request licenses to be shown when the data are to be used. For example, a person's mail server can be modified so that it only accepts an email if a license to use the person's email address is attached in the email. So when a site wants to send advertisements to a person, it must first obtain an agreement from the person. Also, a credit card issuer (such as a bank) can prevent a credit card fraud if an online retailer cannot provide a valid license to use the credit card for a transaction. Note that the license generated by the credit holder contains information to ensure that it can be used only for the transaction, so as to present security attacks such as replay and masquerade.
- Licenses can be used in auditing processes to prevent data misuse. For example, an automatic procedure can be developed to check the licenses of personal data stored at a service provider. The procedure can be integrated into an internal audit system to improve their customers' confidence. Like TRUSTe [5,6], an organization or company can also be established to serve for external auditing to increase service providers' credibility.
- Licenses can be used as evidence to see if a site has used a person's data in the same way as the site requested. In contrast, in passive consent that is generally adopted in the cyberspace, it is very hard for users to prove that a site has used their personal data in accordance with their understandings. This is because a site may change its privacy policies, and a user generally does not keep the copy of the policies he saw when he gave the site his consent.

In this paper we focus on the implementation issues of licenses. Notice that licenses should be made machine-processable so that they can be issued, processed and checked efficiently and automatically. At this point, P3P provides a standard format for expressing privacy practices about personal data. Its rich vocabulary can be used as a foundation to build OPDL. Before presenting the details and necessary modifications, we first discuss some basic privacy principles in the cyberspace. These principles guide how an online service provider should do when it wishes to collect and use customers' data [7,8,9]:

- Notice: The service provider must clearly tell its customers its practices about personal data, e.g., what personal data are to be collected? How the data will be used? Whom they can contact with for complaining? And what choices they have to restrict the collection and use of their personal data?
- Choice: Its customers must have options to decide whether or not to allow the data to be collected and used.

- Individual participation: Its customers must also have right to maintain the correctness of their personal data.
- Security: The service provider must adopt reasonable security safeguards to protect collected personal data.

We can see that P3P meets only the notice and choice principles. P3P does not provide a means for individuals to access and update their personal data at the service providers. This is one of the reasons that the European Union has explicitly rejected P3P as part of its privacy protection framework [10]. Besides, when a user agent receives privacy practices in P3P format (or so called a *P3P policy* in P3P specification [1]) from a Web site, the user agent cannot know what security safeguards the Web site take to protect collected personal data. So P3P also fails to meet the security principle. Because of this, in this paper we focus on how to extend P3P to OPDL in these two directions. Other modifications to P3P that are needed to make OPDL a widely acceptable privacy standard in the cyberspace are discussed in [4].

The rest of the paper is organized as follows. Section 2 reviews P3P and OPDL. In OPDL, requests for personal data are represented as *licensing proposals*. They are based on P3P. In Section 3 we discuss how to make a P3P policy into an OPDL licensing proposal by adding security labels to the policy. Section 4 introduces users' preferences and how OPDL deals with licensing proposals based on the preferences. Section 5 discusses the implementation and verification of licenses. Section 6 presents the mechanism for individuals to update their licenses. Section 7 discusses related work. Finally, conclusions and future work are offered in Section 8.

## 2 Overview

### 2.1 P3P

The first public draft of P3P was proposed by W3C [11] in mid 1998. The original concept can be described as follows: P3P defines a vocabulary and specification for a Web site to make its privacy statements. The privacy statements are represented as machine readable "proposals" to describe what personal data will be collected by the site, for what purposes, other recipients of the data, and the destruction timetable. When a user requests a Web page (to which the user has not yet achieved a privacy agreement) from the site, a set of proposals is sent to the user. The user's agent can then choose one proposal that matches the user's preferences, and sends an agreement ID back to the site to express acceptance of the proposal. After receiving the agreement, the site will transfer the requested page to the user. If none of the proposals is accepted by the user, the site can send another set of proposals for further negotiation.

In the above process, the Web site may also request the user's data. This feature was originated from Open Profiling Standard (OPS) [12]. OPS was intended to provide privacy protection for personal profile information exchange over the Web, and was folded into the early P3P. If the user accepts a proposal,

the requested data along with the agreement ID are transmitted to the site (in HTTP request header [13]). The automatic transfer of personal data raises some controversies, however. So the P3P Specification Working Group later decided to remove this function [14]. The negotiation module was also simplified due to the complexity of the original process. These two modifications have then established the prototype of current P3P.

### 2.2 OPDL

Generally speaking, OPDL acts as a delegate for data subjects (the owners of data) to generate licenses to use their personal data. As depicted in Fig. 1, when a requester requests a person's data, the request is sent through the *Data Licensing Interface*. Requests are more or less like a proposal, and so we refer to them as *licensing proposals*. The main components of a licensing proposal include the data that are requested, the purposes, and the retention timetable. Note that personal data may refer to dynamic click-streams a person generates while he is surfing a Web site.

The kernel of OPDL is *Personal Data Licenser*. Its implementation is not compelled to any form: e.g., it can be implemented as a software agent in a person's computer, or as service offered by a service provider or even on a peer-to-peer network. The main components of Personal Data Licenser are also shown in Fig. 1. The components include three databases: the person's data that the licenser manages, the preferences the person set on licensing, and the logs of licensing proposals (called *Access Log* in the figure). There are also six functional components. *Agreement Negotiator* deals with requests based on a person's licensing preferences. The *Notification Mechanism* deals with the following two tasks:

- To some sensitive data, data subjects can specify more strict preference policies so that the Notification Mechanism will inform them to let them make the final decision.
- When people wish to modify the content of issued licenses (including the data content), the related Internet services will be notified through their *Update Notification Interface*. The details are discussed in Section 6.

After verifying that a licensing proposal has met a data subject's preferences, the Agreement Negotiator generates a license for requested personal data. To prevent the content of the license from being altered and to ensure that the license is indeed issued by the genuine data subject, the license will be signed with the data subject's digital signature. The signature is required for resolving possible dispute in between the licenser and licensee, for example, about the correctness of the data provided by the licenser.

The license (that also contains the requested personal data) is then sent back to the requester. Licensing proposals and the issued licenses are recorded in Access Log to allow data subjects to trace them through *Administration Interface*, which provides functions for data subjects to set or modify their personal data and preferences. The actual processing of the preferences is done by the *Preferences Processor*.

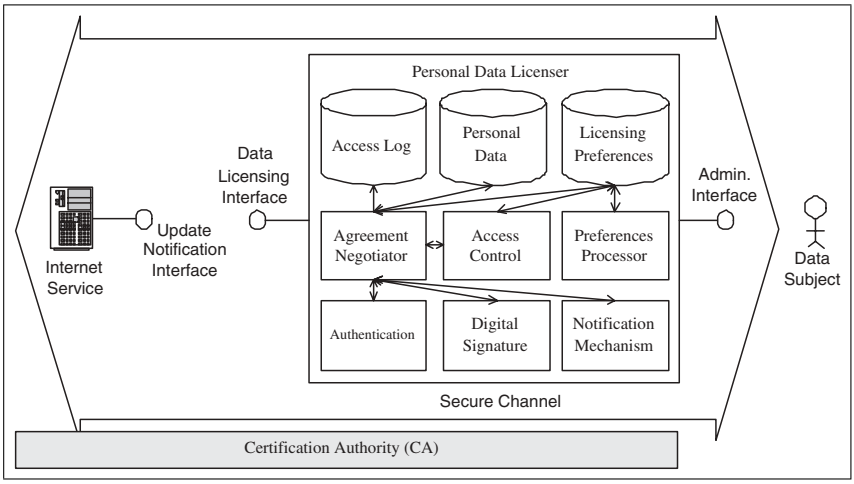



**Fig. 1.** Architecture of Online Personal Data Licensing.

Security mechanisms must be employed to protect the data. The Authentication mechanism verifies users' identities. Access Control checks if requesters have the authorization to access the requested objects. Secure channels can be established to protect communications. For example, a requester can build SSL communications with the Personal Data Licenser, and X.509 certificates can be exchanged for both parties to authenticate each other.

## 3 Security Principle in Licensing Proposals

Recall that the security principle requires service providers to adopt reasonable security safeguards to protect collected personal data. P3P, in general, does not support this principle, and so we need to make some modification in order to adopt it into our licensing proposals. For this, a security evaluation framework is needed for users to evaluate the degree of data security in a service provider. In tradition, several such frameworks have been proposed for this purpose. For example, the Trusted Computer System Evaluation Criteria (TCSEC) [15]. TCSEC is proposed by the U.S. Department of Defense (DoD) to provide a metric to evaluate the degree of trust it can place on computer systems for the secure processing of sensitive information. TCSEC, together with other similar standards like the German Green Book and the British CLEF, was later folded into a single world-wide standard called the Common Criteria in 1994 [16].

The above frameworks are designed to evaluate computer systems. A standard is also required to evaluate and verify the data security of an organization. Some recent emerging standards such as BS 7799 and ISO 17799 enable a person to judge whether or not an organization has defined and put in place effective security processes [17]. (BS 7799 was first announced in 1995 under the auspices of British Standard Institution. It was later replaced by and internationalized

as ISO 17799.) A BS 7799/ISO 17799 compliant service provider should have a well-defined policy about data security. Main security threats (risks) to the data should also be identified, and appropriate controls should be implemented to manage the identified risk.

Of course, a certification organization is needed to let individuals trust the security policy claimed by a service provider. In P3P, Web sites can specify which seal programs they have passed by giving URLs of the pages that contain the validation information. This can also be used for a service provider to specify the certification organization that has verified its security. However, P3P specification does not specify any automatic verification procedure for a user agent to decide the truth of a seal. If a user himself needs to read the validation page for verification, then this would conflict with a design philosophy of P3P—to allow its user to automate decision-making [1]. Therefore, we recommend that there should be a verification interface between certification organizations and user agents. The name of a service provider that needs to be verified can be sent to a certification organization through HTTP GET method. Then, the certification organization can transmit responses (in Web pages) with well-defined structure to the requesting user agent directly.

To illustrate how the above issues are taken into account in licensing proposals, Fig. 2 shows an example of a P3P-based licensing proposal. In this proposal, an online shopping site (exampleshop) wishes to collect some personal data from its members and to use these data for some purposes. It has another version of human readable proposal disclosed in "http://exampleshop/human-readableproposal.html". The proposal is signed by the site to avoid masquerade. In this example, the URLs of the site's security policy, risk assessments, and controls about the risks are offered in the SECURITY-POLICY element so that users can evaluate the data security of the site. In the DISPUTES element, the site states that the proposal is verified by "certification.example.org".

The above security information disclosed by a proposal is generally very difficult for a user to understand. So some characterization about the degree of security safeguards a site uses should be attached in its licensing proposals. The characterization should also be machine-readable so that user agents can process it automatically. For the characterization, we borrow the concepts of divisions in TCSEC to provide individuals a simple yardstick for judging the degree of security safeguards.

In TCSEC, there are four divisions of security, A, B, C, D, among which division A has the highest level of security. Division D is reserved for information systems that cannot meet the requirements of the higher level divisions. The remaining three introduce three concepts about security policies: Division C shows the concept of 'discretionary protection' where an information system must ensure that the access of collected data are accountable, and data and users (the persons that are to use the data, not the owner of the data) are identifiable. Furthermore, audit information about what action is initiated and by whom is kept so that actions affecting security can be traced to the responsible party. Then, division B brings in the concept of 'mandatory protection'. That is, a

⟨POLICY name="example proposal"
discuri="http://exampleshop/humanreadableproposal.html" sigalgorithm="DSA"
signature=" sbxJTQ/YWtQGf75ay/2E6ybTo51gYeInMC0CFQCHapEh+cL14zg5fY
eyl580uj6=" date="2003/3/6 12:34:23 GMT" ID="f1099c8f2f"⟩
⟨SECURITY-POLICY discuri="http://exampleshop/securitypolicy.html"
risks="http://exampleshop/mainrisks.html"
controls="http://exampleshop/riskcontrol.html"⟩
⟨POLICY-TAG⟩⟨DISCRETIONARY /⟩ ⟨/POLICY-TAG⟩⟨/SECURITY-POLICY⟩
⟨DISPUTES-GROUP⟩
⟨DISPUTES resolution-type="independent"
service="http://www.certification.example.org"
verification="http://www.certification.example.org/verify?entity=exampleshop&
proposalid=f31e0e5fea" short-description="certification.example.org"⟩
⟨REMEDIES⟩⟨correct/⟩⟨/REMEDIES⟩ ⟨/DISPUTES⟩
⟨/DISPUTES-GROUP⟩
⟨STATEMENT⟩ ⟨CONSEQUENCE⟩ The id/password used to login and access our
Web site.⟨/CONSEQUENCE⟩
⟨PURPOSE⟩⟨individual-decision/⟩⟨/PURPOSE⟩
⟨RETENTION⟩⟨stated-purpose/⟩⟨/RETENTION⟩
⟨DATA-GROUP⟩⟨DATA ref="#user.login.id"/⟩
⟨DATA ref="#user.login.password"/⟩ ⟨/DATA-GROUP⟩
⟨/STATEMENT⟩
⟨STATEMENT⟩⟨CONSEQUENCE⟩ To secure and improve our Web site. To determine
your habits, interests, or other characteristics for the purpose of research, analysis,
reporting, generating recommendations, and tailoring our Web site.
⟨/CONSEQUENCE⟩
⟨RETENTION duration="2y"⟩⟨stated-purpose/⟩⟨/RETENTION⟩
⟨PURPOSE⟩⟨admin/⟩⟨develop/⟩⟨tailoring/⟩⟨individual-analysis/⟩⟨individual-decision⟩
⟨/PURPOSE⟩
⟨DATA-GROUP⟩⟨DATA ref="#dynamic.clickstream"/⟩ ⟨/DATA-GROUP⟩
⟨/STATEMENT⟩
⟨STATEMENT⟩⟨CONSEQUENCE⟩ To determine your habits, interests, or other
characteristics for the purpose of research, analysis,reporting, generating
recommendations, and tailoring our Web site.⟨/CONSEQUENCE⟩
⟨PURPOSE⟩⟨stated-purpose/⟩⟨individual-analysis/⟩⟨individual-decision⟩⟨/PURPOSE⟩
⟨RETENTION⟩⟨indefinitely/⟩⟨/RETENTION⟩
⟨DATA-GROUP⟩⟨DATA ref="#user.bdate"/⟩⟨DATA ref="#user.gender"/⟩
⟨/DATA-GROUP⟩
⟨/STATEMENT⟩
⟨STATEMENT⟩ ⟨CONSEQUENCE⟩ We will send you some advertisements you might
be interested in.⟨/CONSEQUENCE⟩
⟨PURPOSE⟩⟨contact/⟩⟨telemarketing/⟩⟨/PURPOSE⟩
⟨RETENTION⟩⟨stated-purpose/⟩⟨/RETENTION⟩
⟨DATA-GROUP⟩ ⟨DATA ref="#user.name"/⟩ ⟨DATA ref="#user.home-info.postal"/⟩
⟨DATA ref="#user.home-info.telecom.telephone"/⟩
⟨DATA ref="#user.home-info.online.email"/⟩⟨/DATA-GROUP⟩
⟨/STATEMENT⟩
⟨/POLICY⟩

**Fig. 2.** An example of licensing proposal based on P3P.

division B-compliant information system must enforce a set of mandatory access control rules. Finally, division A is characterized by the use of 'formal security verification' methods to assure that the security controls employed by the information systems can effectively protect personal data used by the information systems.

The above features are used by OPDL and represented as tag elements of DISCRETIONARY, MANDATORY, FORMAL-VERIFIED, respectively. The tags are attached to the SECURITY-POLICY element based on what division level the site has met. For example, in Fig. 2 the site claims to have achieved the discretionary protection.

The other information in the example expresses the site's practices about personal data. This is put in the STATEMENT elements. In the example, the site needs id/password to authenticate its members. The information is retained until its members decide to opt-out (stop using the service). The site will trace its users' click-streams to secure and improve the site. The click-streams will also be used to determine their habits, interests, or other characteristics for the purpose of research, analysis, reporting, generating recommendations, and tailoring the site. These data will be destructed after two years. Similarly, some demographic data, such as birthday and gender, will also be used to provide tailored services. Finally, the site will use its members' address, telephone and e-mail address to contact with them and to provide some recommendations and advertisements.

## 4 Automatic Proposal Processing

To automate proposal processing in OPDL, users can set their privacy preferences so that software agents can decide whether or not to accept a proposal based on the privacy preferences a user set. The privacy preferences are presented in rules based on APPEL [18]. This section describes preference rules, and discusses how the rules can be used by a Personal Data Licenser to process a proposal. For example, Fig. 3 shows a person's privacy preferences setting. The example contains four preference rules: the first rule allows requesters to obtain the person's birth date, gender, and job title; the second rule says that a proposal will be rejected directly if the proposal is not verified by *trustcertorg* or the requested data are not to be protected under a discretionary security policy; the third rule allows 'TrustOrganization' to collect the person's click-stream for an administration purpose and the data will not be retained after the stated purpose is vanished; the fourth rule says that all other information requests should inform the person to allow him to make the final decision.

In general, the components of a preference rule can be categorized into two types: (1) the actions taken when a proposal matches the rule, and (2) one or more expressions. The actions of a rule are (a) reject or accept a proposal directly, and (b) inform the user to make the final decision. An expression can be viewed as a 'predicate' asserting some subjects in an incoming licensing proposal. The possible subjects include:

```
⟨RULESET⟩
⟨RULE behavior = "request" prompt ="no"⟩
   ⟨POLICY⟩⟨STATEMENT⟩
   ⟨DATA-GROUP connective= "or"⟩
      ⟨DATA ref = "#user.bdate"/⟩
      ⟨DATA ref = "#user.gender"/⟩
      ⟨DATA ref = "#user.jobtitle"/⟩
   ⟨/DATA-GROUP⟩
   ⟨/STATEMENT⟩⟨/POLICY⟩
⟨/RULE⟩
⟨RULE behavior = "block" prompt ="no"⟩
   ⟨POLICY connective = "non-or"⟩
      ⟨POLICY-TAG⟩⟨DISCRETIONARY /⟩⟨/POLICY-TAG⟩
      ⟨DISPUTES-GROUP⟩
            ⟨DISPUTES service="http://trustcertorg"/⟩⟨/DISPUTES-GROUP⟩
   ⟨/POLICY⟩
⟨/RULE⟩
⟨RULE behavior = "request" prompt ="no"⟩
   ⟨POLICY⟩
      ⟨ENTITY⟩⟨DATA ref="#business.name"⟩TrustOrganization⟨/DATA⟩
      ⟨/ENTITY⟩
      ⟨STATEMENT⟩
         ⟨PURPOSE connective="or-exact"⟩⟨admin/⟩⟨/PURPOSE⟩
         ⟨RETENTION connective="or-exact"⟩⟨stated-purpose/⟩⟨/RETENTION⟩
      ⟨/STATEMENT⟩
   ⟨/POLICY⟩
⟨/RULE⟩
⟨RULE behavior = "limited" prompt = "yes"⟩
   ⟨OTHERWISE/⟩
⟨/RULE⟩
⟨/RULESET⟩
```

**Fig. 3.** A user's privacy preferences setting.

- Data targeted by the rule, such as a user's gender (with name #user.gender), birthday (with name #user.bdate), or job title (with name #user.jobtitle).
- The target data requester.
- The requirement on a certification organization.
- The security requirement.
- The constraint on request purposes.
- The constraint on retention policies.

An expression is evaluated to TRUE/FALSE depending on if the 'predicate' asserted by the expression is true or not. For example, an expression ⟨DATA ref = "#user.gender"/⟩ is TRUE if a licensing proposal does request the user's gender. If a rule contains more than one expression, the evaluated values are logically combined based on the *connective* attribute of the rule. A proposal matches a rule if the result of the rule is TRUE.

Finally, Fig. 4 shows how a Personal Data Licenser processes a proposal based on a user's preference rules. Basically, if any one of the rules blocks the proposal, then the proposal is rejected. A notification form will be generated

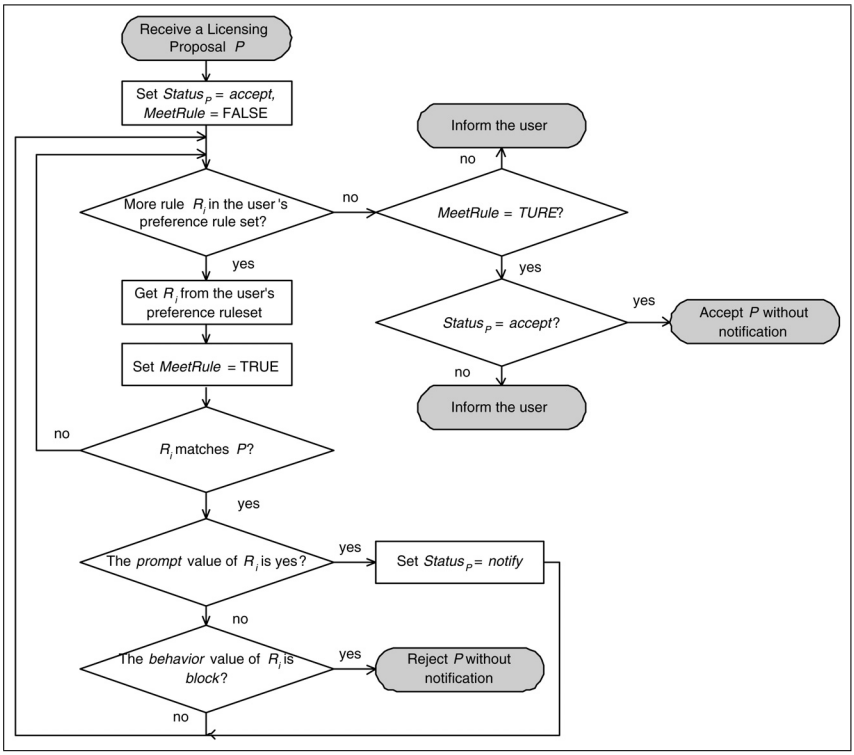



**Fig. 4.** Proposals processing.

and sent to the user if the action of a matching rule is to notify the user (in the *prompt* attribute of the rule). The details about a notification form are given in [4]. Moreover, a notification form will also be generated if no preference rule has been set by the user.

## 5 Licenses

This section discusses the design and implementation issues of licenses. Fig. 5 shows the logical view of a license. Suppose that a license is issued from a person $X$ (with private key $SK_X$) to a service provider $Y$. The main components of the license include a header $H$ and a number of clauses $C_1, \ldots, C_n$. The header contains the general information about the license, e.g., the licenser, the licensee, the time the license is issued, and the security level claimed by the licensee. The clauses describe allowed practices about personal data. Clauses are generally independent, meaning that one clause can be extracted from the license without effecting the integrity of another. Extracted clauses can also be treated as a license (by appending an appropriate header). This decomposable feature has the following benefits:

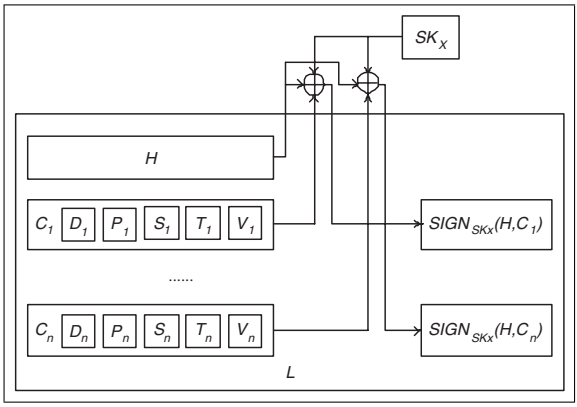



**Fig. 5.** Logical view of a license.

- An audit or verification mechanism may only need part of the license, and so there is no need for a service provider to present the whole license. For example, when using email address of a user, the user's mail server may only need to verify if the service provider has the license to use his email address. Providing only the necessary part of a license speeds up the verification process, and may also protect the privacy of the licenser.
- If a person wishes to update some part of his issued license, the person can update related clauses (see Section 6) instead of reissuing the whole license.

For each clause $C_i$, it contains the allowed privacy practices about a set $D_i$ of data items; that is, the purpose $P_i$ for using the data, with whom ($S_i$) the information may be shared, and the destruction time $T_i$ of the data (or validation period of the clause). In addition, the person may also assign a set of values $V_i$ to $D_i$. (Some data item may not have an initial value to be provided by the licenser, e.g., behavior logs and click streams).

Finally, to prove that the license is indeed issued by the person and that clauses are not altered, each $C_i$ together with the license header $H$ is signed with the licenser's private key $SK_X$. Therefore, the clauses can be verified separately as discussed above.

Licenses can be implemented based on the vocabulary of P3P. An example of a license based on the proposal in Fig. 2 is given in Fig. 6. As shown in this figure, a license includes the following components:

First of all, the whole license is stored in an XML document. Each license has a unique ID as its name, wdich is composed of the licenser's unique identity and the timestamp when the license is issued. The name will then be used for update notification.

In the header of the license, ISSUER and HOLDER show the licenser and the licensee, respectively. ISSUE_DATE records the time the license is issued. It can then be used to calculate the validation period of the license. Furthermore,

SECURITY-LEVEL describes the security level the licensee adopts to protect the target data in this license.

License clauses are represented as CLAUSE elements in the LICENSE_BODY. Basically, other than some validation information, the CLAUSE element has the same structure as the STATEMENT element in the licensing proposal. The signature of each clause is encoded in BASE64 format and is attached as attributes in the clause for verification. Note that an XML document usually has multiple representation forms. For example, an empty element (e.g., ⟨CONSEQUENCE /⟩) can also be represented as a start-end tag pair (e.g., ⟨CONSEQUENCE⟩⟨/⟩). Therefore, contents of licenses must be canonicalized [19] while licenses are signed and verified. Finally, the default values of data items can be set as contents of the corresponding DATA elements in the license.

## 6 Update Notifications

To meet the individual participation principle, individuals can inform licensees of the update to the contents of their licenses. To achieve this, OPDL-compatible services should provide an Update Notification Interface for receiving the notifications of updates so that individuals can send update notifications to them.

The detailed process of an update notification can be depicted in Fig. 7. If a person $X$ wishes to update his license issued to a service provider $Y$, he can send an update request $U_X$ to the service provider through his Personal Data Licenser. The request may belong to one of the following types:

- addLicenseClause($L_{ID}$, $C_{new}$, $SIGN_{SK_X}(H, C_{new})$): To add a new clause $C_{new}$ into a license with ID = $L_{ID}$. The content of the new clause $C_{new}$ and its signature $SIGN_{SK_X}(H, C_{new})$ is also transmitted. These data can be appended to the original license because clauses of a license are mutually independent.
- deleteLicenseClause($L_{ID}$, $C_{ID}$): To invalidate a clause $C_{ID}$ in a license.
- updateClause($L_{ID}$, $C_{ID}$, $C_{new}$, $SIGN_{SK_X}(H, C_{new})$): To update the content of a clause in a license.
- withdrawLicense($L_{ID}$): To withdraw a license.

The P3P vocabulary can be used in the same way as in licenses to express an update request, and so a detail demonstration is skipped here.

After receiving an update request, the service provider signs its response $R_{U_X}$ and the update request with its private key $SK_Y$. The response along with the signature $SIGN_{SK_Y}(R_{U_X}, U_X)$ are transmitted back to the person. As in data licenses, the signature can be used to solve disputes between the person and the service provider about the genuineness of the update request.

In some situations, updates to licenses must be guaranteed, e.g., the revocation of a credit card. A malicious service provider may deny an update request by ignoring response to the request. A Third-Party Notification Complaint Center is therefore required here to deal with the complaints from individuals and

⟨LICENSE ID="CSC_f1099c5ea6"⟩ ⟨LICENSE_HEADER⟩
⟨ISSUER⟩⟨NAME⟩Shi-Cho Cha⟨/NAME⟩⟨CERT_SERIAL⟩12345678
⟨/CERT_SERIAL⟩ ⟨/ISSUER⟩
⟨ISSUE_DATE⟩Fri Jul 14 16:16:58 GMT+8:00⟨/ISSUE_DATE⟩
⟨HOLDER⟩⟨NAME⟩exampleshop⟨/NAME⟩
⟨CERT_SERIAL ⟩87654321⟨/CERT_SERAL⟩⟨/HOLDER⟩
⟨SECURITY-REQUEST⟩⟨DISCRETIONARY /⟩⟨/SECURITY-REQUEST⟩
⟨/LICENSE_HEADER⟩
⟨LICENSE_BODY⟩
⟨CLAUSE sigalgorithm="DSA" signature="MC0CFQCHapEh+cL14In5fYeyl580uj6
tcwIUP/ZVsOFg64zw/1F7 waTo51gWFzg=" ID="f1099c8f2f"⟩
⟨CONSEQUENCE⟩ The id/password used to login and access our Web site.
⟨/CONSEQUENCE⟩
⟨PURPOSE⟩⟨individual-decision/⟩⟨/PURPOSE⟩
⟨RETENTION⟩⟨stated-purpose/⟩⟨/RETENTION⟩
⟨DATA-GROUP⟩⟨DATA ref="#user.login.id"⟩ shichoc ⟨/DATA⟩
⟨DATA ref="#user.login.password"/⟩⟨/DATA-GROUP⟩
⟨/CLAUSE⟩
⟨CLAUSE sigalgorithm="DSA" signature="MC0CFQCWsmkv6kpEKCeKhVswPDb
R9TN7GgIUP3k8rDZPlahL0QkCv0BizABFeiI=" ID="f10b83dbb9"⟩
⟨CONSEQUENCE⟩ To secure and improve our Web site. To determine your
habits, interests, or other characteristics for the purpose of research, analysis,
reporting, generating recommendations, and tailoring our Web site.
⟨/CONSEQUENCE⟩
⟨RETENTION duration="2y"⟩⟨stated-purpose/⟩⟨/RETENTION⟩
⟨PURPOSE⟩⟨admin/⟩⟨develop/⟩⟨tailoring/⟩⟨individual-analysis/⟩
⟨individual-decision⟩⟨/PURPOSE⟩
⟨DATA-GROUP⟩⟨DATA ref="#dynamic.clickstream"/⟩
⟨/DATA-GROUP⟩⟨/CLAUSE⟩
⟨CLAUSE sigalgorithm="DSA" signature="MCwCFFKcscvKr5PJApq+zcuSGuqcc6
Z4AhQWxV4hOxAMPKtfZAKhZPTd6qZbYQ==" ID="f11e0955e5"⟩
⟨CONSEQUENCE⟩ To determine your habits, interests, or other characteristics
for the purpose of research, analysis, reporting, generating recommendations,
and tailoring our Web site.⟨/CONSEQUENCE⟩
⟨PURPOSE⟩⟨stated-purpose/⟩⟨individual-analysis/⟩⟨individual-decision⟩
⟨/PURPOSE⟩
⟨RETENTION⟩⟨indefinitely/⟩⟨/RETENTION⟩
⟨DATA-GROUP⟩⟨DATA ref="#user.bdate"/⟩⟨DATA ref="#user.gender"⟩
Male ⟨/DATA⟩⟨/DATA-GROUP⟩
⟨/CLAUSE⟩
⟨CLAUSE sigalgorithm="DSA" signature="MCwCFHzXv5zI9wTqd5Dbg+wkr1tG7A
opAhQVaaK88Wq8JJYGTAi6472MFc8SCw==" ID="f19068a0e6"⟩
⟨CONSEQUENCE⟩ We will send you some advertisements you might be
interested in.⟨/CONSEQUENCE⟩
⟨PURPOSE⟩⟨contact/⟩⟨telemarketing/⟩⟨/PURPOSE⟩
⟨RETENTION⟩⟨stated-purpose/⟩⟨/RETENTION⟩
⟨DATA-GROUP⟩ ⟨DATA ref="#user.name"⟩ Shi-Cho Cha ⟨/DATA⟩
⟨DATA ref="#user.home-info.postal"/⟩
⟨DATA ref="#user.home-info.telecom.telephone"/⟩
⟨DATA ref="#user.home-info.online.email"⟩csc@eland.com.tw⟨/DATA⟩
⟨/DATA-GROUP⟩
⟨/CLAUSE⟩⟨/LICENSE_BODY⟩ ⟨/LICENSE⟩

**Fig. 6.** A license based on the licensing proposal in Fig. 2.

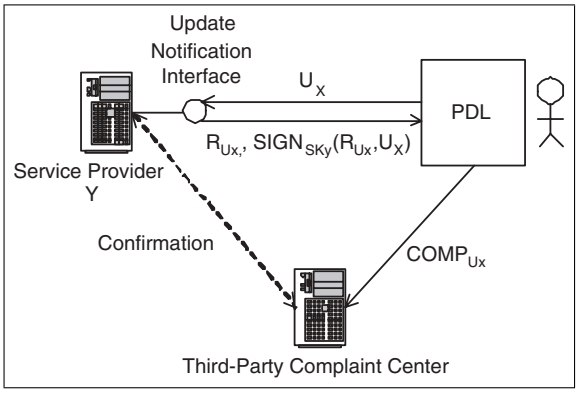



**Fig. 7.** Update notifications.

to distinguish the situation in which the service provider's Update Notification Interfaces is transiently unavailable, from the situation in which the service provider has ignored the update request on purpose. To do so, when a person does not receive the response to its update request from the service provider after a number of attempts, the person sends his complaint $COMP_{U_X}$ about the update request to the Notification Complaint Center. The Notification Complaint Center then takes the responsibility of confirming the truth of the complaint. Some remedies may be taken if the complaint has been confirmed.

One may wonder who will serve for the complaint service. We first observe that current seal programs also have similar functions. For example, TRUSTe's WatchDog Dispute Resolution Service [5] enables a person to file a complaint about a TRUSTe-approved Web site. If the complaint has been confirmed, TRUSTe may require the site to change its stated privacy policy, to remove the site from the TRUSTe program, or even to sue the site for breach of its contract with TRUSTe. For OPDL, as described before, a third-party certification organization plays an important role in verifying the faithful execution of a site's proposal and its security policy. The organization can also serve for the complaint service.

Note that depending on whether or not licenses can be transferred (directly or indirectly), license revocation may become a complicated process. For example, suppose a service provider $Y$ has a license $L_y$ that is dependent on a license $L_x$ hold by $X$. Then, a revocation of $L_x$ at $X$ may incur the revocation of $L_y$ at $Y$. That is, a revocation may trigger a sequence of revocations at different sites. This type of chained revocations will be considered in the future.

A more fundamental question is whether or not we should allow a license to be modified or even withdrawn after it is issued. In fact, the answer may depend on the laws of a country, as well on the content of the license. Here we discuss only the general concept of data licenses. Their applications to specific circumstances are left to the future work.

## 7 Related Work

Digital Right Management (DRM) technologies [20] (e.g., Windows Media Right Manager (WMRM) [21] and IBM's Electronic Media Management System (EMMS) [22]) have recently been developed as a protection against the illegal distribution of copyrighted online publications such as music, e-books and videos. The technologies embed copyright information into online publications so that users must obtain licenses in order to read the publications. The publications may also be encrypted so that they can only be accessed based on the keys in the licenses. Obviously, licenses play an important role to allow a system to automatically check whether or not end users have permissions to access the content of a publication. Several standards of licenses have also been designed for this purpose, e.g., the eXtensive Media Commerce Language(XMCL) [23] (used by IBM's EMMS), and the eXtensive rights Markup Language (XrML) [24] (used by Microsoft's WMRM).

Both DRM and OPDL aim to regulate the use of data. However, because their targets are different (DRM on digital media, while OPDL on personal data), there are differences between the two. In particular, the types of elements in personal data are far more complex than that of digital media. In fact, personal data may contain digital media (e.g., a person's portrait or his self-introduction voice), but not vice versa. Because of this, licenses in OPDL need to deal with more issues than licenses in DRM, especially on privacy. For example, a license in OPDL may involve several elements (e.g., email address, credit card number, pictures), and different applications may need only to verify the license of some of the elements. Therefore, the decomposition feature of licenses proposed in Section 5 is important in protecting personal data from being unnecessarily exposed to different verification mechanisms. In contrast, a license in DRM is usually for a whole, non-decomposable digital publication.

Moreover, because of the complexity of personal data, several issues need to be taken into account when issuing a license, including, for example, the purpose of using the data, security levels of the requesters, and update notifications. So a licensing proposal needs to be presented and verified before issuing a license.

## 8 Conclusions and Future Work

This paper discusses the implementation issues about OPDL. Generally speaking, OPDL concretizes a user's consent on allowing another people to use his personal data by letting the user to issue his own licenses to use the data. Compared to P3P, OPDL not only lets individuals know the privacy practices of a Web site, but can also legally enforce the practices.

Our future work includes the following directions. First, the success of OPDL relies on legislation: *giving the licenses the evidence power in law and requiring non-owners of personal data to obtain licenses from their owners before collecting, processing, and using the data*. How the laws should be made requires deliberate integration between information technologies and jurisprudence, and remains an important and interesting future work.

Second, we observe that OPDL focuses on the collection and process of personal data in the Internet. Internet, however, is not the only source of personal data for enterprises to collect. For example, enterprises may request their customers to fill in paper with their personal data for communication or other purposes. Other forms may also be used in the physical world. This brings up an integration problem about the exchange and use of licenses in different formats, which will be considered in the future.

Moreover, because OPDL adopts P3P for users to express their privacy preferences, it also takes the critique about the specification: too complex for users to manage their privacy preferences. Therefore, a friendly user interface or a set of default preference settings is needed to remove this concern.

The current prototype of OPDL we have designed provides only a simple request and reply architecture. More complex features can be added in the future. For example, a negotiation mechanism can be designed for requesters and Personal Data Licenser to reach an agreement both parties can accept. Moreover, personal data may be treated as properties [25]. So a pricing mechanism may be investigated for requesters to pay for the license fee (i.e., to lease personal data). For example, people may charge commercial e-mail senders for using their e-mail address to send advertisements. OPDL also opens up another direction for future work: design and develop applications to incorporate data licenses. For example, email servers can be modified to filter out unlicensed use of email addresses.

## References

1. Cranor, L., Langheinrich, M., Marchiori, M., Presler-Marshall, M., Reagle, J.: Platform for Privacy Preference (P3P). In: W3C Recommendations. (2002) Retrieved from `http://www.w3c.org/TR/P3P/`.
2. EPIC, Junkbuster: Pretty poor privacy: An assessment of p3p and internet privacy. http://www.epic.org/reports/prettypoorprivacy.html (2000)
3. Isenberg, D.: The GigaLaw—Guide to Internet Law. Random House Trade Paperbacks (2002)
4. Cha, S.C., Joung, Y.J.: Online Personal Data Licensing. In: Proceedings of the 3rd International Conference of Law and Technology (LAWTECH2002). (2002) 28–33
5. TRUSTe: Retrieved from `http://www.truste.org`. (2002)
6. Benassi, P.: TRUSTe: an online privacy seal program. Communications of the ACM **42** (1999) 56–59
7. for Economic Cooperation, O., (OECD), D.: Guidelines on the protection of privacy and transborder flows of personal data. Committee for Information, Computer, and Communication Policy (1980)
8. U.S. Federal Trade Commission: Privacy online: a report to congress. Retrieved from: http://www.ftc.gov/reports/privacy3/index.htm (1998)
9. U.S. Department OF Commerce: Safe harbor privacy principles. http://www.export.gov/safeharbor/SHPRINCIPLESFINAL.htm (2000)
10. European Comission: Platform for privacy preferences and the open profiling standard. Draft opinion of the working party on the protection of individuals with regard to the processing of personal data. http://www.epic.org/privacy/internet/ec-p3p.html (1998)

11. World-Wide Web Consortium: W3C publishes first public working draft of P3P 1.0. http://www.w3.org/Press/1998/P3P (1998)
12. Hensley, P., Metral, M., Shardanand, U., Converse, D., Meyers, M.: Proposal for an open profiling standard. In: W3 Consortium (available as `http://www.w3.org/TR/NOTE-OPS-FrameWork.html`). (1997)
13. Kristol, D.M.: HTTP Cookies: Standards, privacy, and politics. ACM Transactions on Internet Technology (TOIT) **1** (2001) 151–198
14. W3C: Removing data transfer from P3P (1999) Retrieved from `http://www.w3c.org/P3P/data-transfer.html`.
15. US Department of Defense: Trusted Computer System Evaluation Criteria. Technical Report 5200.28, US Department of Defense (1985)
16. Kaufman, C., Perlman, R., Speciner, M.: Network Security: Private Communication in a Public World. Prentice Hall (2002) ISBN: 0-13-046019-2.
17. Calder, A., Watkins, S.: IT Governance: Data Security & BS 7799/ISO 17799. Kogan Page Ltd. (2002) ISBN: 0-7494-3845-2.
18. Cranor, L., Langheinrich, M., Zurich, E.: A P3P Preference Exchange Language 1.0 (APPEL1.0). In: W3C Working Draft. (2002) Retrieved 20 Aug. 2002 from `http://www.w3c.org/TR/P3P-preferences.html`.
19. Boyer, J.: Canonical XML. W3C Recommendation Version 1.0, W3C (2001)
20. Sonera Plaza Ltd MediaLab: Digital Rights Management white paper. Technical report, Sonera Plaza Ltd (2002) http://www.medialab.sonera.fi.
21. Microsoft Corporation: Windows Media Rights Manager 9 series - Live DRM. Technical report, Microsoft Corporation White Paper (2002) http://www.microsoft.com/windows/windowsmedia/drm/livedrm.pdf.
22. IBM Corporation: Electronic Media Management System. Technical report, IBM Corporation (2000) http://www-1.ibm.com/industries/media/pdf/emms_brochure_in_english.pdf.
23. Ayars, J.: XMCL - the eXtensible Media Commerce Language. W3c note, W3C (2002)
24. ContentGuard: XrML 2.1 overview. Technical report, ContentGard (2002)
25. Lessig, L.: Code and other Laws of Cyberspace. Basic Books (1999)

# Author Index