Investigation of Network-based Approaches for Privacy

Korba, Larry; Song, Ronggong

https://doi.org/10.4224/8914087

National Research Council Canada
Conseil national de recherches Canada

Institute for Information Technology
Institut de Technologie de l'information

ERB-1091

NRC·CNRC

Investigation of Network-based Approaches for Privacy

Larry Korba and Ronggong Song

November 2001

Canada

NRC 44900

National Research Council Canada | Conseil national de recherches Canada

Institute for Information Technology | Institut de Technologie de l'information

Investigation of Network-based Approaches for Privacy

Larry Korba and Ronggong Song

November 2001

Investigation of Network-Based Approaches for Privacy

Larry Korba Ronggong Song
Institute for Information Technology,
National Research Council of Canada
E-mail: {Ronggong.Song, Larry.Korba}@nrc.ca

Abstract

We first provide an overview of the better known network approaches for assuring anonymity and privacy over networks. We then analyze possible attacks to these network based on traffic analysis techniques and discuss their implementation and design issues for network privacy. In the conclusion we discuss the results, indicating research opportunities in this domain.

1. Introduction

Privacy is becoming a critical issue on the Internet. Users feel that one of the most important barriers to using the Internet is the fear of having their privacy violated. Governments around the world have introduced or are building legislation that places requirements upon the way in which personal information is handled. In attempt to provide some technical solutions to fill the apparent privacy void, several network-based privacy-enhancing technologies have been developed in recent years. Some examples of these technologies include: Dc-Nets [10], MIX network [14, 38], Onion Routing network [16, 33], Crows system [34], Freedom network [39], Pseudonym IP [51], etc.

In this document we provide an overview of the better known network approaches for assuring anonymity and privacy over networks. The key approaches we discuss include: Dining Cryptographer Networks (DC-Nets), JANUS Proxy, MIX Network, Onion Routing, Crowds System, Freedom Network, and NymIP. While these approaches offer some possibilities for providing the anonymity and privacy, they do not answer all of the requirements for agent-based systems. For instance, many of these approaches are designed for World Wide Web access. Being protocol-specific these approaches may require further development to be used with agent-based systems, depending on the communication protocols used in those systems. For the most part the network-based privacy protocols are geared towards client-server configurations. This should not detract from using the protocols for agent-based systems, since the privacy approaches offer bi-directional exchanges. Core to many privacy developments is the use of multiple proxies. Many of the approaches may be compromised if the attacker gains access to multiple nodes of the privacy network. One approach to make this extremely difficult to do is to have the proxies located in different countries. To make these systems practical, approaches are needed especially to handle key distribution. Handling keys can be especially cumbersome in agent systems where there are many hundreds or thousands of agents.

While privacy and anonymity would be core requirements for agent operation, we discuss other requirements for the potential tracking of privacy-related activities for the purpose of enforcement and management. In this case, the need for privacy and anonymity must be tempered by the need for accountability and provability to be compliant with potential privacy auditing requirements. This is an area requires considerable research effort. Zero Knowledge Systems "freedom network" offers a useful Internet infrastructure for privacy that can be useful for demonstrations of Privacy Enhancing Technologies (PET). In addition, there may be research opportunities in the development of alternate approaches for anonymity.

The rest of the paper is organized as follows. The background about network, privacy, agents and privacy-enhancing technologies is briefly reviewed in the next section. In Section 3, some anonymous communication network technologies are reviewed. In Section 4, the possible threats of these networks are analyzed and compared. In Section 5, the implementation and design issues of these networks are concisely described. In Section 6, some concluding remarks and directions are presented for further research.

2. Background

The Internet is currently the focus of a great deal of attention. It has come to be seen as an 'information infrastructure' for every subject and application domain. E-commerce and e-government are becoming part of citizens' everyday life. The tremendous growth of the Internet has catapulted it from its original realm of academic research towards new mainstream acceptance and increasing social relevance and reliance for hoi polloi. This dramatic increase in reliance on the Internet while providing some benefits also has the potential of eroding personal privacy.

The fact is that the cyberspace has invaded private space. Controversies about spam, cookies and the clickstream form merely the tip of an iceberg. It is small wonder that lack of public confidence is a serious impediment to the acceptance rate of consumer e-commerce, e-government, etc. The concerns are not merely about security of value, but about something much more significant: trust in the information society.

Privacy is becoming a critical issue on the Internet. Users feel that one of the most important barriers to using the Internet is the fear of having their privacy violated. Governments around the world have introduced legislation placing requirements upon the way in which personal information is handled. In attempt to provide some technical solutions within the privacy void, several network-based privacy-enhancing technologies have been developed in recent years. Some examples of these technologies include: Dc-Nets, MIX network, Onion network, Freedom network, Pseudonym IP.

On the other hand, computers are commonly used for an increasing range of everyday activities. Most of these activities are based on the exchange and acquisition of information. At present, however users still interactively and directly initiate all actions needed for a computer to execute tasks. With the use of software agents, computer systems are capable of executing actions with minimal supervision by their users. This gives the user more time to spend on other activities. Recently, research and development efforts in the area of agent technologies have increased significantly. It is expected that they will take important role in the future information society and especially in e-business applications. Surprisingly, the research and development of these technologies is going on with little concern for the privacy issues raised by user demand and government regulations.

This document discusses privacy and agents from the network perspective. Through the course of this document we describe research opportunities and methods for building privacy enhancing technologies for agent systems.

2.1 Networks, privacy and agents

- ***Networks***

The Internet comprises a network of computer networks, which transmit messages between entities using a common set of communications protocols, or sets of operating rules. Networks comprise addressable and transparent devices or nodes connected by communication channels.

Nodes can be differentiated according to the services they provide. They include computers such as workstations and servers, and intermediating nodes such as routers, bridges and domain-name servers.

Nodes are not limited to performing a single role. For each of the roles that a particular node performs, it is assigned a unique identifier, called an IP-address, protocol and port number.

The communication channels that connect nodes may be implemented using any of a number of technologies. These include various forms of physical medium, such as copper wire, co-axial cable and fibre-optic cable; and wireless: infrared, electromagnetic transmission either at low altitude, such as microwave links and cellular mobile networks, or via satellite.

The Internet has brought with it enormous benefits. Those benefits relating to virtual communities are being realized; but those relating to e-commerce are delivering on their promise much more slowly. Central among the impediments are the multitudinous negative impacts on privacy. For example, since Internet uses IP-address for providing communication between nodes, some metadata of Internet communication (e.g. email address, IP-address, HTTP cookies) and traffic analysis become significant threats to the personal data. However, the Internet itself doesn't provide protection against these threats. Thus, an anonymous communication network implemented on or embedded in the Internet is vital for protecting the privacy of individual.

- ***Privacy***

Privacy is the interest that individuals have in sustaining a 'personal space', free from interference by other people and organizations. In other words, privacy is the right of a person to be let alone. It has multiple facets. Information Privacy is the interest an individual has in controlling, or at least significantly influencing, the handling of data about themselves.

Privacy refers to the ability of individuals to control the collection, retention and distribution of information about themselves. This doesn't mean that their personal information never gets revealed to anyone; rather, a system that respects their privacy will allow them to select what information about them is revealed, and to whom. This personal information may be any of a large number of things, including their shopping habits, nationality, work history, living habits, personal communications, email address, IP address, physical address and identity.

Privacy protection is a process of finding appropriate balances between privacy and multiple competing interests. Recently, several techniques have been developed for providing privacy protection. Generally, they can be divided into two kinds of techniques: one is for providing pseudonym protection, another is for providing anonymity and unobservability using anonymous communication networks.

- ***Agents***

An agent is a software construction that knows how to do things that you could probably do yourself if you had the time. Depending on their intended use, agents are referred to by a variety of names, e.g., bot, robot, personal assistant, transport agent, mobile agent, search agent, navigation agent, intelligent agent, management agent, domain specific agent, etc.

An essential element of e-commerce and e-government is the collection of information in many databases connected over large computer networks. Currently, the technologies for the retrieval of information on large computer networks, especially the Internet, have become more and more complicated. With the overwhelming information stored in the many workstations and servers, the time and capacity needed for retrieval of information is growing strongly. This results in combining intelligence and agency, that is, the development of Intelligent Software Agent Technologies (ISATs). However, ISATs could present a significant threat to privacy due to the wealth of personal information in their possession and/or under their control. Thus, it is necessary that Privacy-Enhancing Technologies (PET) be implemented especially for the software agents. Interestingly, the PET developing and applied for agent systems will also likely have application to the more traditional client-server Internet communication paradigm.

2.2 Privacy-Enhancing Technologies

In [24] I. Goldberg gives an overview of existing and potential privacy enhancing technologies for the Internet. Based on the implementation techniques and functions, we can classify the network-based privacy enhancing technologies into two kinds: one is for converting user's identity into a pseudonym using pseudonym techniques; another is for hiding the user's identity and providing unobservability against traffic analysis using anonymous communication network techniques. Other privacy enhancing techniques include: information protection within agents, protection of agent code from intentional or accidental damage, secure distributed logs, among others.

- ***Pseudonym Techniques***

In [4, 11, 20, 21, 29] several pseudonym techniques are proposed and developed. The primary goal of pseudonym techniques is for the hiding of the user's identity in the application data, i.e., using pseudonym instead of identity. Of course, pseudonym techniques have other advantages, e.g. authentication, abuse control, accountability, etc.

Pseudonym techniques can be implemented using proxy-based or agent-based approaches. Actually, the JANUS proxy is a pseudonym technique. Our PET-Agent also is a pseudonym techniques. The disadvantage of pseudonym techniques is that they cannot provide the personal data protection against traffic analysis by themselves. This function should be provided with anonymous communication networks.

- ***Anonymous Communication Network techniques***

In [1, 10, 14, 17, 19, 39, 44] some anonymous communication networks are proposed and developed. The primary goal of the anonymous communication network is to protect user anonymous communication against traffic (content and flow) analysis.

Current implementation techniques of anonymous communication networks include Dc-Nets, MIX networks, Onion Routing networks, Crowds systems and Freedom networks. Pseudonym IP is only a proposal. The disadvantage of anonymous communication networks is that they cannot provide user's identity protection in the application data. This function should be provided with pseudonym techniques.

2.3 Conclusion

The PISA project involves the development of privacy enhancing technologies for next generation applications for electronic commerce. Distributed applications are foreseen as the major part of these next generation applications. Agent systems are one type of distributed applications Intercommunicating, multiple agents comprise multi-agent systems. They may communicate using a variety of different networks: wireless, wired; telephone, Internet, intranet, etc. These networks form the substrate upon which agent applications operate. In order for agents to intercommunicate, standards and network protocols have been developed. These protocols support the interaction between agents, but tend not to answer questions concerning the privacy requirements for interactions.

The standard network privacy requirements related to the agent-based electronic commerce include:
- Protection of agent communication
- Anonymity of agent operation

Please note that the protection of private information within agents or databases is considered to be non-network related.

3. Anonymous Communication Approach

Privacy requirements have been recognized for many years. TCP over IP version 4 is designed to allow computers to easily interconnect and to assure that network connections will be maintained even when various links may be damaged. This same versatility makes it rather easy to compromise data privacy in networked applications. For instance, networks may be sniffed for unencrypted packets, threatening the confidentiality of data. Research and development however, have led to techniques to provide private communications between parties. In this section we concisely describe some of the more commonly known network privacy technologies.

3.1 Dining-Cryptographer Networks (DC-Nets)

In [10] D. Chaum describes a technique, the DC-Nets, which should allow the transmission and reception of messages anonymously using an arbitrary communication network. A short and slightly generalized description of the Dc-Nets follows.

Assume that a number of participants want to exchange messages over an arbitrary communication network. A computationally unlimited attacker, who is able to eavesdrop on communication between any two of the participants and who controls an arbitrary subset of the participants, tries to trace the messages exchanged between the participants to their senders and recipients.

The goal here is to have one participant anonymously broadcast a message. If all messages are delivered to each participant, the attacker is not able to trace the intended recipient of a message. Therefore recipient anonymity is trivially maintained.

An anonymous multi-access channel guarantees sender untraceability.

Let $P=\{P_1, P_2, \ldots, P_n\}$ be the set of all participants and let G be an undirected self-loop free graph with nodes P. Let $(F, \oplus)$ be a finite *Abelian group*, in which all computations will be carried out. The set F is called the *alphabet*. The protocol goes as follows.

(1) Initialization: To be able to perform a single sending step, called a round, each pair of participants P_i and P_j who are directly connected by an edge of G choose a key K_{ij} from F randomly. Let $K_{ij}=K_{ji}$. Participants P_i and P_j keep their common key secret. The graph G is called *key graph*, the matrix K of all keys is called *key combination.*

(2) Message Transmission: In order to send a message M, P_i broadcasts:

$$M \oplus \sum_{\forall j,\ \{P_i,\ P_j\} \in G} sign(i-j) \bullet K_{i,j}$$

Where $sign(x)=1$ if $x>1$ and -1 otherwise.

(3) "Noise" Transmission: All other participants, P_j, broadcast:

$$F \sum_{\forall k,\ \{P_j,\ P_k\} \in G} sign(j-k) \bullet K_{j,k}$$

(4) Computing the Message: All interested participants can obtain M by adding ($\oplus$) all broadcasted messages. The fact that the sum of all broadcasted message equals M can be seen by noting that all terms except M cancel out because of *sign()*.

In order to quantify the scheme's security, a graph is defined by having n vertices labeled 1,2,…n, each representing a participant, with edges between nodes i and j if and only if P_i and P_j share a secret key. If the graph obtained by removing the vertices corresponding to the participants controlled by an adversary is connected, then the protocol protects sender anonymity. This analysis is tight in that if the graph isn't connected, an attacker can determine within which of the disconnected parts of the graph the sender is located. This is a theoretical solution but for GPRS-applications not feasible.

3.2 MIX Network

A MIX network takes a list of values as input, and outputs a permuted list of function evaluations of the input items, without revealing the relationship between input and output elements. David Chaum [14] introduced MIX-networks in 1981, in order to enable unobservable communication between users of the Internet. A single MIX does nothing else than hide the correlation between incoming and outgoing messages within a large group of messages. An architecture for MIXes proposed as unobservable Internet access is illustrated in Figure 1 [38].

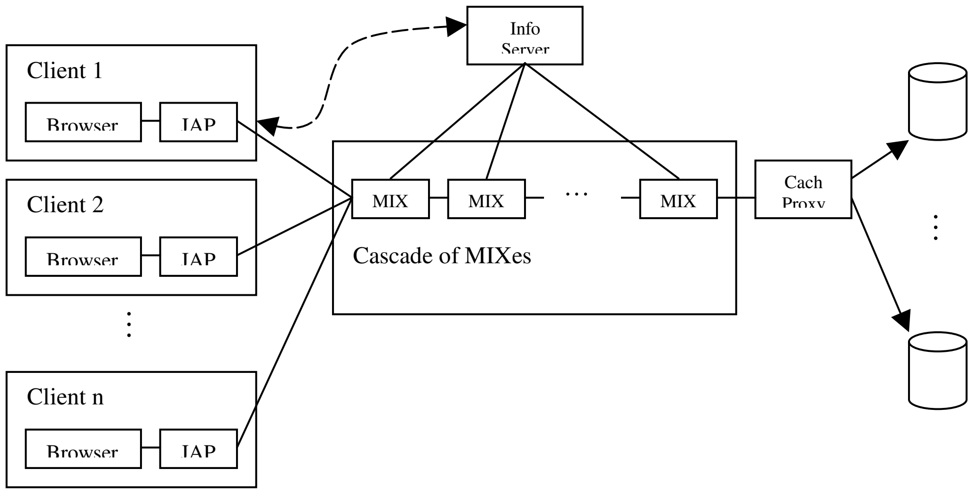

Figure 1. Architecture of MIXes as unobservable Internet access

A MIX node is a processor that takes as input a certain number of messages which it modifies and outputs in a random order. The messages are modified and reordered in such a way that it is nearly impossible to correlate a message that "come in" with a message that "go out". The MIX nodes can be used to prevent traffic analysis in roughly the following manner.

(1) The message will be sent through a series of MIX nodes, say i_1, i_2, ..., i_d. The user encrypts the message with node i_d's key, encrypts the result with node i_{d-1}'s key and so on with the remaining keys.
(2) The MIX nodes receive a certain number of these messages which they decrypt, randomly reorder and send to the next nodes in the routes.

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

Each MIX node knows only the previous and next node in a received message's route. The entry and exit nodes know the sender and recipient of the message respectively. Hence, unless the route only goes through a single node, compromising a MIX node doesn't trivially enable an attacker to violate sender-recipient privacy. When using only one MIX, one must rely upon its security completely. Therefore usually several MIXes are ideally used in a chain. Now, any single MIX does not have all the information needed to reveal communication relations. At worst, a MIX may only know either sender or receiver.

There are different possibilities to organize the cooperation of several MIXes as follows.

- *MIX network[8]:* All MIXes exist independently from each other in the Internet. The routes can be chosen at random, that is, the user chooses i_1, i_2, ..., i_d uniformly at random. This type of cooperation is called a MIX network.

- *MIX cascade[6]:* A single valid chain of MIXes is defined for a group of participants. The route can also be constant. This means that it does not change. In this setting, the attacker knows the entry, exit and intermediate nodes. This kind of cooperation is called a MIX cascade.
- *Other Configurations:* There is a high variety of hybrid combinations of MIX network and MIX cascade. For example, one can think of other ways of choosing routes as follows: A) part of the route could be fixed; B) the route could be chosen at random from a set of pre-determined choices; C) the route could be chosen at random subject to some restriction.

There are different approaches to flushing MIX nodes as follows.

- *Message threshold:* The MIX nodes wait until they receive a certain number of messages before "releasing" all of them at the same time.
- *Message Pool[30]:* The flushing algorithm for Mixmaster has two parameters: the pool size *n* and the probability of sending *p*. The nodes wait until they have *n* messages in their pool at which time, they shuffle the messages and send each one with probability *p* (e.g. if *p*=1 the scheme is identical to the message threshold approach).
- *Stop and Go[18]:* D.Kesdogan, Jan Egner and R.Buschkes give an interesting scheme in which messages wait random times at a nodes before being released. In this setting, the attacker has a probability of success: If an empty node receives a message and does not receive another one before sending the decrypted message, the attacker can easily "follow" the message - routing the message through this node doesn't "help". Perhaps the most interesting contribution is that a statistical analysis is used to determine the probability of this happening.
- *Other:* There are many other reasonable algorithms for doing this, for example MIX nodes could receive a random number of messages and output a constant number of them (using dummy messages to fill the gapes).

3.3 Onion Routing

Conceptual development of Onion Routing as well as a description of the design for the previous system can be found in [17, 33]. Brief description of different aspects of the current design can be found in [16, 40].

The primary goal of Onion Routing is to provide strongly private communications in real time over a public network at reasonable cost and efficiency. A secondary goal is to provide anonymity to the sender and receiver, so that the responder may receive messages but be unable to identify the sender, even though the responder may be able to reply to those messages.

In onion routing, instead of making socket connections directly to responding machine, initiating applications make connections through a sequence of machines called onion routers. The onion routing network allows the connection between the initiator and responder to remain anonymous. It is called an anonymous socket connection or anonymous connection. Onion Routing builds anonymous connections within a network of onion routers, which are, roughly, real-time Chaum MIXes. While Chaum's MIXes could store messages for an indefinite amount of time waiting to receive an adequate number of messages to mix together, a Core Onion Router is designed to pass information in real time, which limits mixing and potentially weakens the protection. Large volumes of traffic can improve the protection of real time MIXes.

Anonymous connections hide who is connected to whom, and for what purpose, from both outside eavesdroppers and compromised onion routers. If anonymity is also desired, then all identifying information must be additionally removed from the data stream before being sent over the anonymous connection.

The basic configuration of the onion routing network topology is illustrated in figure 2.

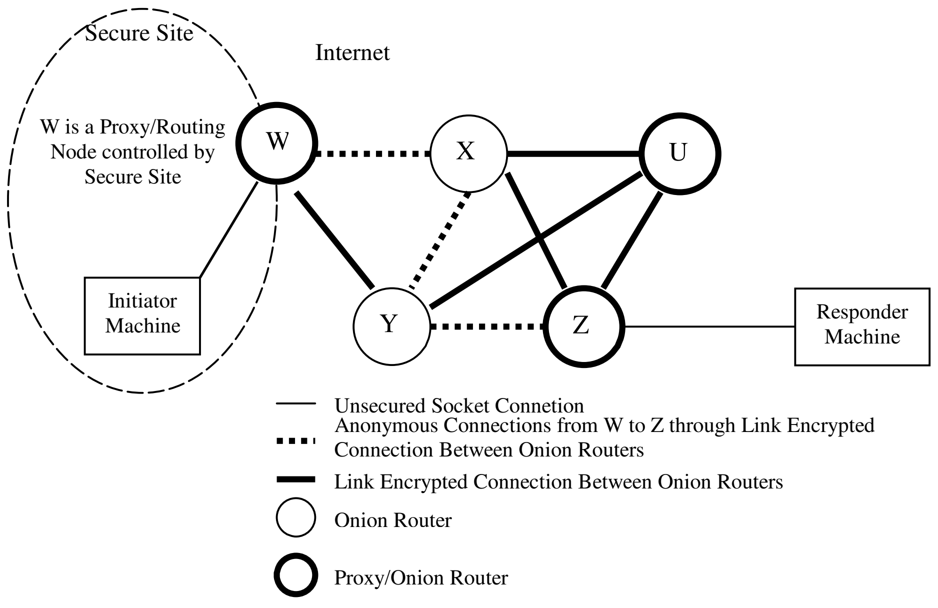
Secure Site
Internet
W is a Proxy/Routing Node controlled by Secure Site
W
X
U
Y
Z
Initiator Machine
Responder Machine
Unsecured Socket Connetion
Anonymous Connections from W to Z through Link Encrypted Connection Between Onion Routers
Link Encrypted Connection Between Onion Routers
Onion Router
Proxy/Onion Router

Figure 2. Routing Topology

In the basic configuration, an onion router sits on the firewall of a sensitive site. This onion router serves as an interface between machines behind the firewall and the external network. Connections from machines behind the firewall to the onion router are protected by other means. To complicate tracking of traffic originating or terminating within the sensitive site, this onion router should also route data between other onion routers.

The use of anonymous connections by two sensitive sites that control onion routers effectively hides their communication from outsiders. However, if the responder is not in a sensitive site the data stream from the sensitive initiator must also be anonymized. Otherwise, even rudimentary analysis of the unprotected communication between the last onion router in the anonymous connection and the responder may reveal the initiator's identity.

Onion routers in the network are connected by longstanding socket connections. Anonymous connections through the network are multiplexed over the longstanding connections. For any anonymous connection, the sequence of onion routers in a route is strictly defined at connection setup. However, each onion router can only identify the previous and next hops along a route. Data passed along the anonymous connection appears differently at each onion router, so data cannot be tracked en route and compromised onion routers cannot cooperate by correlating the data stream each sees.

The onion routing network is accessed via proxies. An initiating application makes a socket connection to an application specific proxy on some onion router. That proxy defines a route through the onion routing network by constructing a layered data structure called an onion and sending that onion through the network. Each layer of the onion defines the next hop in a route. An onion router that receives an onion peels off its layer, identifies the next hop, and sends the embedded onion to that onion router. After sending the onion, the initiator's proxy sends data through the anonymous connection.

The last onion router forwards data to another type of proxy, whose job it is to pass data between the onion network and the responder.

In addition to carrying next hop information, each onion layer contains key seed material from which keys are generated for encrypting and decrypting data sent forward or backward along the anonymous connection.

Once the anonymous connection is established, data can be sent in both directions. The initiator's onion proxy receives data from an application, breaks it into fixed size cell, and encrypts each cell multiple times - once for each onion router the connection traverses - using the algorithms and keys that were specified in the onion. As a cell of data moves through the anonymous connection, each onion router removes one layer of encryption, so the data emerges as plaintext from the final onion router in the path. The responder proxy regroups the plaintext cells into the data stream originally submitted by the application and forwards it to the destination. For data moving backward, from the recipient to the initiator, this process occurs in the reverse order, with the responder proxy breaking the traffic into cells, and successive onion routers encrypting it using different algorithms and keys than the forward path. In this case the initiator's proxy decrypts the data multiple times, regroups the plaintext cells, and forwards them to the application.

By layering cryptographic operations in this way, an advantage over link encryption is gained. As data moves through the network it appears different to each onion router. Therefore, an anonymous connection is as strong as its strongest link, and even one honest node is enough to maintain the privacy of the route. In link encrypted systems, compromised nodes can cooperate to uncover route information.

Although this system is called onion routing, the routing that occurs here does so at the application layer of the protocol stack and not at the IP layer. More specifically, it relies upon IP routing to route data passed through longstanding socket connections. An anonymous connection is comprised of several linked longstanding socket connections. Therefore, although the series of onion routers in an anonymous connection is fixed for the lifetime of that anonymous connection, the route that data actually travels between individual onion routers is determined by the underlying IP network.

Onion routing depends upon connection based services that deliver data uncorrupted an in-order. This simplifies the specification of the system. TCP socket connections, which are layered on top of a connectionless service like IP, provide these guarantees. Similarly, onion routing could easily be layered on top of other connections based services, like ATM.

3.4 Crowds System

Reiter and Rubin [34, 35] propose a lighter weight alternative to MIXes and Dc-Nets. Their system is called Crowds and can be seen as a P2P (peer-to-peer) relaying network in which all participants forward messages. The messages are forwarded to the final destination with probability *p* and to some other participants with probability 1-*p*.

Crowds is a system for protecting user privacy while she/he browses the web. The goal of Crowds is to make browsing anonymous, so that information about either the user or what information he or she retrieves is hidden from Web servers and other parties. Crowds prevents a Web server from learning any potentially identifying information about the user, including the user's IP address or domain name. Crowds also prevents Web servers from learning a variety of other information, such as the page that referred the user to its site or the user's computing platform.

The approach is based on the idea of "blending into a crowd", i.e., hiding one's actions within the actions of many others. To execute web transactions in this model, a user first joins a **crowd** of other users. The user's initial request to a web server is first passed to a random member of the crowd. That member can either submit the request directly to the end server or forward it to another randomly chosen member. In the latter case the next member independently chooses to forward or submit the request. When the request is eventually submitted, it is submitted by a random member, thus preventing the end server from identifying its true initiator. Even crowd members cannot identify the initiator of the request, since the initiator is indistinguishable from a member that simply passed on a request from another.

A proxy running on your local machine (or another machine you trust) executes the Crowds protocol on your behalf. Participating in a Crowd simply requires that you start this proxy and set your browser to use it as the browser's proxy. Thus, any request coming from the browser is sent directly to the proxy. Upon receiving the first user request from the browser, the proxy initiates the establishment of a random path of proxies that carries its user's transactions to and from their intended web servers. More precisely, the proxy picks a proxy from the crowd at random, and forwards the request to it. When this proxy receives the request, it flips a biased coin to determine whether or not to forward the request to another proxy (i.e. the coin indicates whether or not to forward with probability *p*). If the result is to forward, then the proxy selects a random proxy and forwards the request to it. Otherwise the proxy submits the request to the end server for which the request was destined. So each request travels from the user's browser, through some number of proxies, and finally to the end server. A possible set of such paths is shown in Figure 3. In this figure, the paths are 1→5→server; 2→6→2→server; 3→1→6→server; 4→4→server; 5→4→6→server; and 6→3→server. The server replies the same path as the requests, only in reverse.

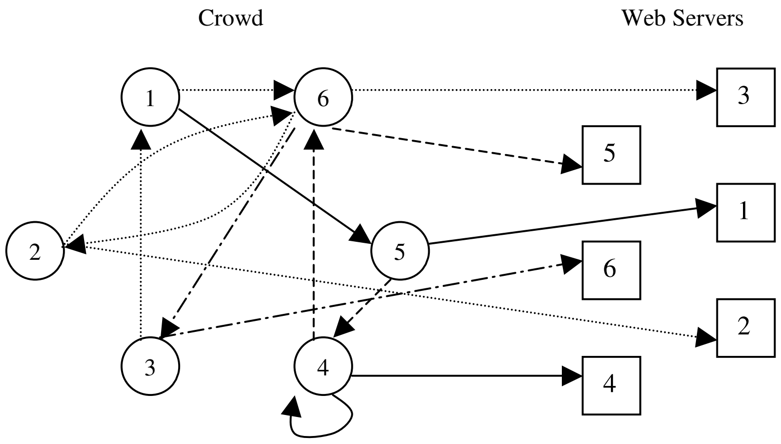

Figure 3. Paths in a crowd (the initiator and web server of each path are labeled the same)

3.5 Freedom Network

This is a MIX network that provides unobservable and anonymous real-time connections between network nodes. So it provides access to different Internet services. The way it works is very similar with the service Onion-Routing. The Freedom Network has been in operation for about 3 years, with anonymizing proxy servers located in ISPs throughout the world. The current product offers anonymizing service for a few protocols. Due to internal issues, Zero Knowledge Systems has not collaborated on this report. Most of the description contained below has been extracted from the whitepapers available at http://www.freedom.net

The Freedom Network is an overlay network that runs on top of the Internet. It uses layers of encryption to allow a Freedom user to engage in a wide variety of pseudonymous activities, hiding the user's real IP address, email address, and other identifying information from eavesdroppers and active attempts to violate the user's privacy. Users are encouraged to create pseudonyms ("nyms") for each area of activity in which they want to preserve their privacy. The nyms that someone uses cannot be tied together. Thus, it is not possible to know if superman@freedom.net and clarkkent@freedom.net are the same or different people. Superman is happy with this situation because he doesn't want his super villain enemies to know about his life. Similarly, when jobseeker@freedom.net browses a resume web site, his employer cannot see that Clark ("jobseeker") isn't happy with his job at the Daily Planet and wants to work elsewhere. Freedom protects Clark's privacy by proxying the various supported protocols, and sending those proxied packets through a private network before they are deposited on the Internet for normal service. That private network, as a system, is operated by Zero-Knowledge. Individual nodes in the network are operated by Zero-Knowledge or their partners, so that no single operator has comprehensive knowledge of what data is flowing through the network. Thus, the main components of the system are Freedom Clients and Freedom Servers. In the next section, provides a more precise definition of these and other entities.

3.5.1 Freedom Network Layout

The Freedom System consists of a set of Freedom Server Nodes that make up the Freedom Network, and the Freedom Core Servers that provide basic services (see Figure 4). The network transports encrypted IP traffic from one node to the next. Nodes on the Freedom Network are called Anonymous Internet Proxies (AIPs). The number of nodes used in a route is chosen by the user by setting her security level in the Freedom client. The server nodes themselves are not linked by a fixed topology, instead, they can communicate with any other server node on the network, as requested by a client when creating a route. The Freedom client is given a network topology that identifies a set of reliable links between nodes in order to simplify route selection. This topology is defined solely on the basis of AIP-AIP performance characteristics.

Authenticated routes to an exit AIP are granted through route creation requests signed by a nym. Upon receipt of such a request, an AIP will retrieve the requesting nym's public signature key and verify the request. Only nyms that have full access to the network can create such a route. Authenticated routes allow access to any host on the Internet.

Unauthenticated routes to a core AIP are granted to any Freedom client that requests one. These are limited in that only Freedom core servers can be accessed.

The core servers consist of PKI server clusters, reporting and network information server clusters, a token server, and the mail system. The core servers are accessed by creating completely anonymous routes that terminate at a core AIP. These are the only AIPs on the Freedom Network capable of accessing the core servers.

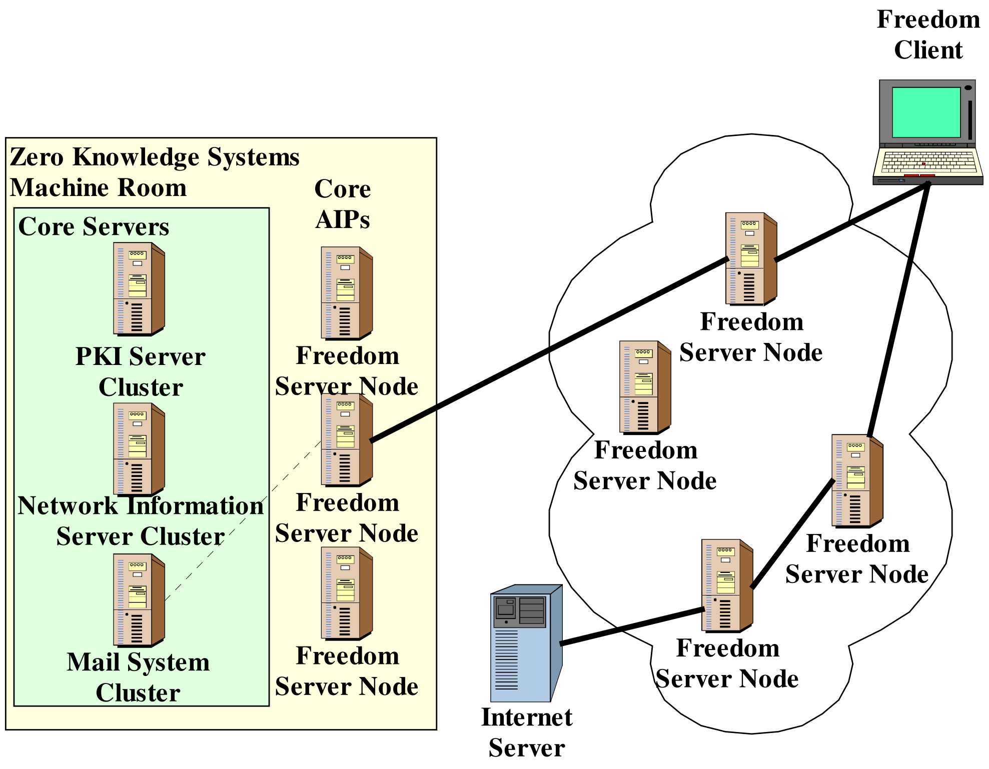

Figure 4. Freedom System

3.5.2 Freedom Network Server Nodes

A Freedom server node is a host running part of the Freedom Network. This node can be administered by Zero-Knowledge or by a third party server operator. A Freedom server node consists of an AIP, a local NIQS daemon client, some administration utilities and a set of key management utilities. The node's operator generates the signature and encryption keys and submits the public components to Zero-Knowledge Systems for distribution to the rest of the network. Zero-Knowledge Systems never sees the node's private keys, ensuring that only that node is capable of decrypting its layer from the traffic that goes through it.

The server node software needs to be able to communicate with the key query servers, NIQS, NISS, and other AIPs on the Freedom Network. It accepts connections from Freedom clients wanting to transport encrypted IP traffic on the Freedom Network.

A Freedom server node also runs a DNS server that is used to respond to DNS queries generated by a Freedom client. This is done through the same authenticated route as the client that issued the query, thus ensuring that any hostname lookup done by a Freedom user pseudonymous and private.

3.5.3 Freedom Core Systems

The Freedom Core Systems provide the base services that keep the Freedom Network running. This includes providing public keys, nym creation and update, network information and reporting, token creation, and the mail system. All of these services are currently provided by Zero-Knowledge and are located within our machine room. This new architecture introduces the concept of a core AIP that allows unauthenticated connections to any of the core servers in the network. The servers are no longer constrained by performance limitations or affected by the AIP running on the same host, as they were in the previous version of the system. Multiple instances of each core server type are also now supported. This allows the Freedom System core to load balance incoming requests across several machines as required.

3.5.4 Software Components

The Freedom System components that make up the architecture design, include the Freedom client, AIP, network information and reporting system, PKI servers, and the mail system.

- ***Freedom Client***

The Freedom client allows networking applications running on a computer to access the Internet through the Freedom System. It is designed to work seamlessly with these applications by trapping and redirecting the data streams that they generate. What follows is a simplified description of the client that applies to both the Windows and Linux versions. A more thorough description is beyond the needs and scope of this document.

The Freedom client consists of several components: a graphical user interface, a network access layer, a traffic filter, application filters, and a set of libraries that implement the functionality required for encryption, routing, nym management, route creation, etc.

The Freedom Mail System is closely linked to the SMTP and POP3 application filters. On top of the sanitization work done by these filters, they also encrypt, decrypt, reconstruct and scan messages, using the appropriate keys as required by the selected nym. This allows the Freedom client to seamlessly integrate a user's preferred mail application into the Freedom System.

- ***Masquerading AIP***

The Anonymous Internet Proxies (AIPs) are the core network privacy daemons that make up the Freedom Network. They pass encapsulated network packets between themselves until they reach an exit node. The masquerading AIP is incorporated into the Linux kernel in order to access the kernel's networking features. This allows the AIP to route large amounts of network traffic efficiently. A user-level daemon handles reliability-demanding operations such as key negotiations.

The Masquerading AIP obtains network information describing nodes and servers within the Freedom System from an NIQS server. It uses the key query server in order to lookup nym public signature keys that are used to sign authenticated route creation requests. The AIP will periodically report its status to a Freedom NISS server.

The capacity of the Freedom Network can be scaled up simply by adding new AIPs. Since there is no topology, each new AIP can be fully used to support new users.

- ***Network Information and Reporting***

The Freedom System's network information and reporting capability is made up of nodes that contain three different components. There is an NISS server to receive status updates from server entities in the Freedom System, a network information database (NIDB) to store the information, and an NIQS server that mines this database to answer network information queries for all entities in the Freedom System. As status updates arrive at the Freedom Core, they are replicated and a copy is sent to each network information node.

The Freedom client does not receive real-time status information on the Freedom Network. Instead, it maintains its own statistics regarding the state of the network based on its accesses to it.

- ***PKI Systems***

The PKI servers consist of three types of server, a key query server, a nym server, and token server. A nym server is used to create and manage a Freedom client user's pseudonymous identities. A key query server is used to provide public keys on request to Freedom servers and clients. The token server provides nym creation tokens, which allow a user to anonymously create nyms without identifying their real identity.

A Key Query Server provides access to public signature and encryption keys for all entities on the Freedom Network. It is used by all entities in order to validate signatures and keys that they receive in the course of their activities.

The token server is used to redeem activation codes for nym creation and upgrade tokens. This transaction is done anonymously through the Freedom Network. This layer of indirection in the nym creation process ensures that a nym cannot be tracked back to its owner when it is created.

The PKI systems interact with most other components of the Freedom System. AIPs lookup public signature keys for nyms that sign route creation requests. The nym server creates mail access certificates that are returned to the client and are in turn passed to the mail system to create nym mail accounts. The Freedom client uses the nym server to create and modify new nyms. The PKI servers themselves obtain network information from the network information system's NIQS server and report their status to the NISS.

3.5.5 Database Systems

The Freedom databases are accessed through the PKI and Network information servers. Any response generated by one of these servers will always be signed by the server's signature key.

- ***Nym Database***

The nym database holds all publicly available information that describes each nym. None of the stored information reveals the true identity of a nym owner. This database can be queried by anyone, but only the owner of a nym is allowed to modify a nym record. When a nym is created, that entry in the nym name space becomes permanent. If the owner of a nym decides to destroy their nym, or the nym expires, that nym name is not returned to the name space and it cannot be used by another Freedom user.

- ***Public Keys***

The public key database holds the public signature and encryption keys for all entities in the Freedom System, including nyms and servers. All keys are stored signed, according to the Freedom key hierarchy.
When a public key lookup is made by a Freedom entity, it needs to lookup the keys used to sign the request key in order to verify the validity of the response. These extra keys are cached by the requesting entity in order to reduce extra key lookups.

- ***Spent Tokens***

The spent token database is used to prevent a nym creation or upgrade token from being reused. When an attempt to spend a token at a nym server is made, the database is searched for a hash of the token. If it is found, the token has already been used and is therefore refused. If it is not found, the hash is stored in the database and the token is accepted. The hash of the token is not linked to any other piece of information.

- ***Activation Code Database***

The token server employs the activation code database to keep track of activation codes that can currently be redeemed by users. Each activation code has an associated capability code that is used to determine the type and number of tokens that are generated when the activation code is redeemed. There is no token database. These are generated and signed as required by the token server.

- ***Network Information Database (NIDB)***

This database maintains information regarding all Freedom server entities. An entity record can be queried independently, or a complete compressed description of the entire network can be queried.

- ***Reporting Database***

This database is maintained by the NISS using the status updates that are sent to it from the various Freedom servers. It keeps track of things such as the uptime of each server, the number of bytes transferred, the number of messages processed, etc. This database is used to manage the Freedom System.

3.6 NymIP

This is an effort to bring about a new standard for pseudonymity and anonymity services for the Internet. The intention is for the developments here to become IETF standards. A number of well known network security researchers have been involved in the effort. The web site for the effort is located at http://nymip.velvet.com/ . While there was a flurry of activity when this effort began in the fall of 2000, the group appears to be dormant (see mailing list archivers at http://nymip.velvet.com/pipermail/nymip-res-group/). Currently the web site contains a list of things that may be desirable in controlled nymity networks at the IP layer.

Zero Knowledge Systems has provided considerable input to the effort. For example, in [37, 51] a framework is proposed for the NymIP. Once again, these are but ideas, no implementations have as yet come from the NymIP effort.

4. Analysis of Anonymous Communication Network Threats

In this section, we describe attacks to some anonymous communication networks. We particularly focus upon those attacks that may reduce the privacy of the individuals. Most of these attacks are using traffic analysis techniques, for example, message coding attack, timing attack, message volume attack, flooding attack, intersection attack, communication pattern attack, collusion attack and denial of service attack, etc. These threats are described as follows.

- ***Message Coding Attack:*** An attacker can easily link and trace some messages if the messages do not change their coding during transmission.
- ***Timing Attack:*** An attacker can observe the set of messages coming in the network and the set of message going out of the network, and get some useful route timing information in correlating the messages in the two sets.
- ***Message Volume Attack:*** An attacker can observe the amount of transmitted data (e.g. the message length, number of messages).
- ***Flooding Attack:*** An attacker may flood a system in order to separate a certain message.
- ***Intersection Attack:*** An attacker may trace some users by observation over a long period because of the on-line/off-line periods of the users.
- ***Communication Pattern Attack:*** An attacker may find out a lot of useful information by simply looking at the communication patterns when users send and receive messages.
- ***Collusion Attack:*** A corrupt coalition of users or parts of the system may be able to trace some certain users.
- ***Denial of Service Attack:*** An attacker may obtain some information about the routes used by certain users by rendering some in-operational nodes.

In addition, there also exist some special attacks for each anonymous communication network.

4.1 Dc-Nets Threats

The Dc-Nets allows transmission and reception of messages anonymously using an arbitrary communication network. It has been proven to be unconditionally secure. Unfortunately, in the Dc-Nets, the transmission of a specific participant *X* can be traced by an active attacker who is able to alter the message received by *X* and who controls the current communication partner of *X*. The active attack on untraceability is as follows [36].

The power of an active attack is based on a very simple observation: for services using two-way communication it is impossible to realize unconditional sender untraceability without unconditional recipient untraceability and vice versa.

Assume that one of the participants controlled by the attacker, say P, communicates with some honest participant X, and that X answers a message M by sending a message M'. If the attacker is able to identity the sender of M', he can identity the recipient of M and vice versa. If the attacker doesn't control P, the same is true for light traffic; then the attacker can identify both communication partners. In general if sending and receiving is correlated (which is usually the case) the attacker can always learn something about recipients from identifying senders and vice versa.

If active attacks are possible, superposed sending doesn't guarantee recipient untraceability and therefore it doesn't guarantee sender untraceability:

Let I_i be the input character which participant P_i receives and which should always be equal to the global sum S.

Assume that the attacker is able to deliver an arbitrary character I^*_i to each participant P_i instead of the correct character I_i. This may be possible, e.g. if the Dc-Nets is implemented using a star whose center collaborates with the attacker. Further assume that participant P, who is controlled by the attacker, communicates with the honest participant X according to some protocol. P knows that X will always answer to a received message M within a given time by sending a message M'.

If the attacker delivers message M consecutively to a single participant only, and a meaningless message to all others, he can always identify X by checking whether he receives M' or not. Instead of delivering M to a single participant only, he can deliver it to a subset of the participants. By successively partitioning the participants he can identity X in $\log(n)$ rounds, provided that the protocol between X and P consists of at least $\log(n)$ interactions.

One could argue that this attack can be avoided by using networks for which it is physically more difficult to manipulate the input characters of all participants, e.g. rings.
But even if the attacker can only manipulate what single participant P_i receives, at least the unobservability of this participant is essentially decreased, since the attacker can test whether he is communicating with P_i or not.
To protect against active attacks, we need to rely on physical devices because secure and reliable broadcast mechanisms can't be constructed by algorithmic means. This problem can be partially fixed by the technique of Waidner [36] that uses a fail-stop broadcast.

In addition, the protocol has the following serious drawbacks.

- *Channel jamming:* If many participants try to send a message at the same time, all is lost as the sum of all broadcast values will equal the sum of the message (e.g. $M_1 \oplus M_2 \oplus \ldots \oplus M_j$). An even bigger problem is if a participant acts maliciously and deliberately sends channel jamming messages; this allows him to compute the legitimate message while the other users can't gain any information.
- *Number of messages:* Every user needs to participant every time a message is broadcasted. This is a problem both in terms of efficiency and robustness. This is an unrealistic constraint in large networks.
- *Shared Secret Keys:* The number of keys to share could be too large for practical purposes (need a new key for each transmission). Pseudo-random numbers can be used to alleviate this problem. The fact that many users need to share secret keys with participants is also a serious problem in terms of practicality.

Despite these problems, the Dc-Nets is useful in many situations (e.g. [43]). The Dc-Nets can also be used in conjunction with other sender-recipient privacy protecting mechanisms such as the MIX network. For example, a certain number of users can transmit information to a MIX network using a Dc-Net. In this setting, even if the MIX network security is violated, the attacker can only ascertain that the sender a given message is one of the parties using the Dc-Net.

4.2 MIX Network

A comprehensive listing of attacks against MIX-networks is discussed in [3, 9, 14, 15, 23, 27, 28, 33, 36, 38, 45].

- **Message Coding Attack**
  The MIX network provides protection using public key cryptography against message coding attack.

- **Timing Attack**
  An attacker having access to just one of the communicating parties might be able to infer which route is taken by simply computing the round trip time. That is, calculating the time it takes to receive a reply. This attack is interesting in that even if one of the parties uses "constant link padding" the attack is still effective.

  The attack motivates the use of MIX nodes that wait variable amounts of time before flushing messages. Randomly increasing the latency doesn't completely solve the problem since an attacker might be able to rule out some routes (e.g. if a message exits the MIX network faster than the minimum time needed to go through some routes then these routes can be ruled out). Hence, the minimum time needed to go through each route should be the same. This kind of attack is mentioned in [27, 33]

  The Chaum's MIX network provides protection against timing attack, but it causes high delay. Recently, the MIX network has been improved by means of dummy traffic and chop-and-slice algorithm.

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

  Recently, the MIX network has been improved against message volume attack by means of dummy traffic and chop-and-slice algorithm.

- **Intersection Attack**
  The MIX network doesn't provide protection against intersection attack.

  An attacker having information about what users are active at any given time can, through repeated observations, determine what users communicate with each other. This attack is based on the observation that users typically communicate with a relatively small number of parties. For example, the typical user usually queries the same web sites in different sessions. By performing an operation similar to an intersection on the sets of active users at different times it is probable that the attacker can gain interesting information. The intersection attack is a well known open problem and seems extremely difficult to solve in an efficient manner.

- **Communication Pattern Attack**
  By simply looking at the communication patterns, one can find out much useful information. Communicating participants normally don't "talk" at the same time, that is, when one party is sending, the other is usually silent. The longer an attacker can observe this type of communication synchronization, the less likely it's just an unrelated random pattern.

  A passive adversary that can monitor entry and exit MIX nodes can mount this attack. Law enforcement officials might be quite successful mounting this kind of attack as they often have a-priori information: they usually have a hunch that two parties are communicating and just want to confirm their suspicion.

- **Packet Counting Attack**
  These types of attacks are similar to the other contextual attacks in that they exploit the fact that some communications are easily distinguished from others. If a participant sends a non-standard number of messages, a passive external attacker can spot these messages coming out of the MIX-network. In fact, unless all users send the same number of messages, this type of attack allows the adversary to gain non-trivial information.

  A partial solution is to have parties only send standard number of messages but this isn't a viable option in many settings.

  The packet counting and communication pattern attacks can be combined to produce a "message frequency" attack. Communication pattern, packet counting and message frequency attacks are sometimes referred to as traffic shaping attacks and are usually dealt with by imposing rigid structure on user communications.

- **The Node Flushing Attack (Flooding Attack, *n*-1 Attack)**
  The flush attack is first mentioned in [14], which is very effective and can be mounted by an active external adversary. If the nodes wait until they have t messages before "flushing", an attacker can send t-1 messages and easily associate messages leaving the node with those having entered. Note that the adversary will be able to match his inputs with the messages leaving the node.

  Dummy traffic can make things a bit more difficult for the attacker since he can't distinguish them from legitimate messages. Unfortunately, if dummy traffic is only used in specific instances, an attacker can choose his messages so that dummy traffic isn't used.

  Another potential solution is to authenticate each message that allows nodes to detect flushing attempts. Unfortunately, this entails authenticating each message and detecting flushing attempts. This approach could be computationally infeasible.

  Stop-and-Go MIX nodes can partially solve this problem. Unfortunately, they only have "probabilistic" security.

  There are some similarities with denial of service attacks [15]. Hence, if t is very large, using hashcach [2] or pricing functions [9] might be effective solutions.

  The Chaum's MIX network doesn't provide protection against flooding attack, but the MIX network has been improved recently by means of a "ticket" in [38].

- **Collusion Attack**
  The MIX network provides protection against collusion attack, but it is not secure against intersection attack.

- **Denial of Service Attack**
  By rendering some MIX nodes inoperative, an active adversary might obtain some information about the routes used by certain users. It seems highly probable that users that have their routes "destroyed" will behave differently than parties that haven't. Network unobservability or "client challenges" might be required to properly solve this problem for real-time interactive communication.

In addition, the MIX network has some threats caused by the following attacks.

- **Brute Force Attack**
  The brute force attack is very instructive because it can help in determining how much, where and when to use dummy traffic. Dummy messages are messages that are sent through the network in order to complicate the attacker's task. On the Internet, MIX network operators sometimes need to pay for each message and so we want to be sure the dummy messages have a good security to cost ratio.

  The idea is very simple: follow every possible path the message could have taken. If the MIX isn't well designed and the attacker is extremely lucky, he can link sender and recipient. In most cases however, the attacker will be able to construct a list of possible recipients.

  The attack can be carried out by passive, static and external adversaries, and capable of tapping the required wires. If the attacker cannot tap a relevant wire, he won't be able to produce a complete potential recipient list since the paths going through the missing wires, are lost. A passive, external, adaptive adversary is better suited to this problem as he can "follow" the messages, compromising only the relevant channels.

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

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

- **Message Manipulating Attack**
  If an attacker changes only some bits of a message, then if inappropriate cryptography is used he can detect this broken message also after it has passed a trustworthy MIX. This is possible especially if:
  - the attacker is the receiver of the message, or
  - he controls the server, which would send the decrypted message to the real receiver.

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

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

- **Message Blocking Attack**
  If an attacker blocks some messages, the senders of these messages are excluded from the anonymity group. The same result is achieved, if some messages are manipulated as above.

  The extreme case is the "*n*-1 Attack". In this case all messages but one, will be blocked, manipulated or generated by the attacker. The only message the attacker doesn't know is the message he wants to trace. In this way the attacker has bridged the trustworthy MIX or has limited the anonymity group of the remaining messages.

  There exists no general applicable method, in order to prevent this attack. A possible solution is that the MIX must be able to identify each user. The MIX ensures that enough different users send the messages it receives and so the attacker doesn't control a majority of them.

- **Active Attacks Exploiting User Reactions**
  Active adversaries might be able to gain non-trivial sender-recipient matching information by exploiting the fact that user behavior depends on the message received. A variation on the following attack can even be used against a Dc-Nets network that is not using a secure broadcast channel.

  1. The adversary first intercepts a message *M* just before it enters the MIX network.
  2. *M* is then sent to a set of possible recipients. The parties not expecting to receive this message will probably react in a different manner than a party expecting it.

  The attacker can use this to get some sender-recipient matching information. If the nodes in the route authenticate the messages the attack is prevented.

- **The "Sting" Attack**
  If one of the parties involved in a dialog is corrupt, he might be able to, in a sense, "encode" information in his messages [27]. For example, government agencies might set up a fake "bomb making instruction web sites" and try to find out who accesses it. Many methods for identifying a user querying the web page come to mind: varying the reply latency, sending messages of a specific length, etc.

  In some situations, it might be even easier to compromise user privacy. For example, if the sting web site gives fake information pertaining to financial fraud, the user might act upon this information at which point he can be arrested via an “out of channel” interaction.

- **Attacks Based on the Message's Distinguishing Feature**
  If the user's messages have distinguishing characteristics, the recipient might be able to link the message with one or many individuals. For example, analyzing writing styles probably reveals non-trivial information about the sender.

- **Message Delaying Attack**
  The attacker can withhold messages until he can obtain enough resources or until the network becomes easier to monitor or to see if the possible recipient receives other messages, etc. In view of this attack, it makes sense to have the MIX nodes verify authenticated timing information.

- **Message Tagging Attack**
  An active internal adversary that has control of the first and last node in a message route, can tag messages at the first node in such a way that the exit node can spot them. Since the entry node knows the sender and the exit node the recipient, the system is broken.

  An active external adversary can mount a slight variant of this attack if the messages don't have a rigid structure. This attack has many similarities with subliminal channels [22]. This observation forms the basis of some of the following variations:

  Shadow Messages: If an adversary sends messages that follow the same path as the message being followed, it can easily transmit some information to the output. For example, the attacker can just replay the message in such a way that it can spot them leaving the MIX network.

  Message Delaying: The attacker can delay messages to obtain some information. These delays can presumably be detected.

  Broadcast: An attacker can broadcast messages notifying his accomplices that a particular message has entered the network. This isn't a particularly powerful attack but it could be virtually impossible to detect.

  The message tagging based attacks motivate using extremely rigid message structure and authenticating timing information.

- **Partial or Probabilistic Attack**
  Most of the preceding attacks can be carried out partially, that is, the attacker can obtain partial or probabilistic information. For example, he could deduce information of the form:
  1. With probability *p*, *A* is communicating with *B* or *A* is communicating with one of the users in a group.
  2. *A* is not communicating with *B, C* and *D*.

  These attacks haven't been thoroughly addressed so far and seem very promising, especially when carried out a large number of times.

A MIX network cannot be realized securely, especially when we think of the well known attacking model defined by D. Chaum. The theoretical anonymity group of a very big MIX network, that contains all users, is practically not achievable, even if all MIXes are trustworthy ones.

4.3 Onion Routing

Like the MIX network, the Onion Routing does not provide protection between end-users against timing attacks, intersection attacks and flooding attacks, etc. With enough data, it is possible to analyze usage patterns and make educated guesses about the routing of messages. Also, since Onion Routing requires real time communication, it may be possible to detect the near simultaneous opening of socket connections on the first and last proxy servers revealing who is requesting what information.

- **Message Coding Attack**
  The Onion Routing provides protection against message coding attack using public key cryptography.

- **Timing Attack**
  The Onion Routing provides protection between Onion Routers against timing attacks, but it doesn't provide protection at endpoints against timing attacks.

  A sophisticated adversary may be able to detect timing coincidences such as the near simultaneous opening of connections. Timing coincidences are very difficult to overcome without wasting network capacity, especially when real-time communication is important.

- **Message Volume and Bit Pattern Attack**
  In the Onion Routing network, all data (e.g. onions, content and network control) is sent through the Onion Routing network in uniform-sized cells (at present, 128 bytes). It is encrypted (or decrypted) as it traverses each node, a cell changes its appearance (but not its size) completely from input to output. This prevents an eavesdropper or a compromised onion router from following a packet based on its bit pattern as it moves across the Onion Routing network.

  But the Onion Routing only provides protection between the Onion Routers against message volume attack, and doesn't provide protection at endpoints against message volume attack.

- **Intersection Attack**
  Like the MIX network, an attacker having information about what users are active at any given time can, through repeated observations, determine which users communicate. The Onion Routing doesn't provide protection against intersection attack.

- **The Flooding Attack**
  In order to prevent an eavesdropper from relating an outbound packet with an earlier inbound one based on timing or sequence of arrival, all data cells arriving at an onion router within a fixed time interval are collected and reordered randomly before they are sent to their next destinations. If traffic levels are low and requirements for real-time transmission are high, waiting for enough traffic to arrive so that mixing provides good hiding might cause unacceptable transmission delays. In this case, padding (synthetic traffic) can be added to the thick pipes. Conversely, an attacker might try using a pulse of traffic to track cells flowing through the system. This attack can be made more difficult by imposing limits on the traffic flow over particular links, but this strategy can increase latency.

  Generally, the Onion Routing doesn't provide protection against flooding attack.

- **Collusion Attack**
  The Onion Routing provides protection against collusion attack, but it is not secure against traffic analysis.

- **Replay and Denial of Service Attack**
  The Onion Routing uses expiration times to prevent replay attacks. It is curious that, unlike timestamps, the vulnerability due to poorly synchronized clocks here is a denial of service attack, instead of a replay attack. If a node's clock is too fast, otherwise timely onion will appear to have already expired. Also, since expiration times define the window during which nodes must store used onions, a node with a slow clock will end up storing more information.

  In general it is sufficient for a single routing node to be uncompromised to complicate traffic analysis. However, a single compromised routing node can destroy connections or stop forwarding messages, resulting in denial of service attacks.

- **Compromised Proxy Attack**
  If the initiator's proxy is compromised then all information is revealed. If the responder's proxy is compromised, and can determine when the unencrypted data stream has been corrupted, it is possible for compromised nodes earlier in the virtual circuit to corrupt the stream and ask which responder's proxy received uncorrupted data. By working with compromised nodes around a suspected initiator's proxy, one can identify the beginning of the virtual circuit.
  In order to make the Onion Routing effective, there must be significant use of all the nodes, and Proxy Nodes must also be intermediate routing nodes.

- **Contextual Attack**
  These are the most dangerous attacks and, unfortunately, they are very difficult to model in a rigorous manner. The problem is that real-world users don't behave like those in the idealized model. This class of attack is particularly effective for real-time interactive communication. For example, by simply looking at the communication patterns, one can find out a lot of useful information. Communicating participants normally don't "talk" at the same time, that is, when one party is sending, the other is usually silent. The longer an attacker can observe this type of communication synchronization, the less likely it's just an unrelated random pattern.

The Onion Routing network only prevents attacks from external observers and isolated attacking onion routers. If some onion routers of a route cooperate with the attacker, the attacker has a very good chance to observe a user.

4.4 Crowds System

The Crowds System also does not provide protection between end-users against timing attacks, intersection attacks, flooding attacks and denial-service attacks. Unlike the MIX network and the Onion Routing network that provide sender and receiver unlinkability against a global eavesdropper, the Crowds System does not provide anonymity against global eavesdroppers. The Crowds system protects only against observations done by the communication partner (i.e. the WEB-server) or done by an other, isolated participant.

- **Message Coding Attack**
  The Crowds system provides protection against message coding attack with a certain probability from insiders and in any case from outsiders.

- **Timing Attack**
  The Crowds system doesn't provide protection against timing attack, but request and response bursts are suppressed.

The probability of timing attack in the Crowds system results from the structure of HTML, the language in which web pages are written. An HTML page can include a URL (e.g., the address of an image) that, when the page is retrieved, causes the user's browser to issue another request automatically. It is the immediate nature of these requests that poses the greatest opportunity for timing attacks by collaborating crowd members. Specifically, the first collaborating crowd proxy on a path, upon returning a web page on that path containing a URL that will be automatically retrieved, can time the duration until it receives the request for that URL. If the duration is sufficiently short, then this could reveal that the collaborator's immediate predecessor is the initiator of the request.

In the present implementation, the Crowds system eliminates such timing attacks. The disadvantage of the present approach to defending against timing attacks is that it is not easily compatible with some web technologies. For example, web pages contain executable scripts (e.g. written in JavaScript). In addition, misbehavior by the last crowd proxy on the path can result only in a denial of service, and not in a successful timing attack.

- **Message Volume, Intersection and Flooding Attacks**
  The Crowds system doesn't provide protection against message volume attack, intersection attack and flooding attack.

- **Denial of Service Attack**
  The Crowds system makes no effort to defend against denial of service attacks by rogue crowd members. A crowd member could, for example, accept messages from other crowd members and refuse to pass them along. Such denial of service can result from malicious behavior, but typically does not result if a crowd member fails benignly or leaves crowd. As a result, these attacks are detectable. More difficult to detect are active attacks where crowd members substitute wrong information in response to web requests that they receive from other crowd members. Such attacks are inherent in any system that uses intermediaries to forward unprotected information, but fortunately they cannot be used to compromise anonymity directly.

In addition, the Crowds system has the following threats.

- **Sender Anonymity Attack**
  The Crowds System serves primarily to hide the sender or receiver from the attacker. It makes no effort to hide correlations between inputs to and outputs from the initiating computer. That is, when the user initiates a request, the fact that she did so is exposed to the local eavesdropper. Thus, the local eavesdropper observes that a request output by the user's computer did not result from a corresponding input. So the Crowds system doesn't offer sender anonymity against a local eavesdropper.

  However, the anonymity for the path initiator is quite strong against an attack by the end server. Since the path initiator first forwards to another Crowd proxy when creating its path, the end server is equally likely to receive the initiator's requests from any crowd member. That is, from the end server's perspective, all crowd members are equally likely to have initiated the request, and so the actual initiator's anonymity is beyond suspicion.

  The Crowds system can also offer sender anonymity against collaborating crowd members. Consider any path that is initiated by a non-collaborating member and on which a collaborator occupies a position. The goal of the collaborators is to determine the member that initiated the path. Assuming that the contents of the communication do not suggest an initiator, the collaborators have no reason to suspect any member other than the one from which they immediately received it, i.e., the member immediately preceding the first collaborator on the path. All other non-collaborating members are each equally likely to be the initiator, but are also obviously less likely to be the initiator than the collaborators' immediate predecessor.

- **Receiver Anonymity Attack**
  Because the web server is the receiver, obviously receiver anonymity is not possible against an attack by the end server.

  However, the Crowds system typically prevents a local eavesdropper from learning the intended receiver of a request, because every message forwarded on a path, except for the final request to the end server, is encrypted. Thus, while the eavesdropper is able to view any message emanating from the user's computer, it only views a message submitted to the end server if the user's crowd proxy ultimately submits the user's request itself. Since the probability that the user's crowd proxy ultimately submits the request is 1/n where n is the size of the crowd when the path was created, the probability that the eavesdropper learns the identity of the receiver decreases as a function of crowd size. Moreover, when the user's crowd proxy does not ultimately submit the request, the local eavesdropper sees only the encrypted address of the end server.

  The Crowds system can also offer receiver anonymity against an attack by collaborating crowd members.

Finally, the most important disadvantage of the Crowds system is that it cannot support TLS and Firewalls. Thus, this limits its application for most of e-commerce environments.

4.5 Freedom Network

The Freedom network also does not provide protection between end-users against timing attacks, intersection attacks, flooding attacks and denial-service attacks.

- **Message Coding Attack**
  The Freedom network also provides protection against message coding attack using public key cryptography.

- **Timing Attack**
  The Freedom network also doesn't provide protection at endpoints against timing attack, however it is planned to provide protection against timing attack using traffic shaping algorithm.

- **Message Volume Attack**
  The Freedom network also doesn't provide protection at endpoints against message volume attack, however it is planned to provide protection against message volume attack using traffic shaping algorithm.

- **Intersection and Denial of Service Attack**
  The Freedom network doesn't provide protection against intersection attack and denial of service attack.

- **Collusion Attack**
  The Freedom network provides protection against collusion attack, but it is not secure against traffic analysis.

In addition, A.Back gave more information about Freedom 2.1 security issues and analysis in [1].

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

4.6 NymIP

This is a new activity that does not yet have an implementation, let alone a detailed analysis of the privacy/security threats.

5. Implementation and Design Issues

5.1 Dc-Nets

Based on the above analyses of Dc-Nets threats, some practical issues related to Dc-Nets are considered here.

- **Fail-Stop Broadcast instead of Reliable Broadcast**

  In [10] the untraceability of senders and recipients of messages is proved to be unconditional, but this proof implicitly assumes a reliable broadcast network. The problem of achieving reliable broadcast on a network which does not provide it automatically, is also known as the Byzantine Generals problem. Its solution by protocols is known as the Byzantine Agreement [31, 32]. Since unconditional Byzantine Agreement is impossible, such a network cannot be realized by cryptographic means. Thus the assumption may be rather unrealistic.
  In [36] a fail-stop broadcast is proposed. The fail-stop broadcast combines both advantages: it can be implemented in a more efficient way than reliable broadcast and it is unconditionally secure in spite of arbitrary attackers. By choosing the keys necessary for the Dc-Nets dependently on the previously broadcast messages, the fail-stop broadcast can be achieved with unconditional security and without increasing the complexity of the Dc-Nets significantly, using an arbitrary communication network.

- **Security and Authentication**

  A sender can ensure the secrecy of an anonymous message by encrypting the message with the intended recipient's public key. The sender can even keep the identity of the intended recipient secret by leaving it to each recipient to decrypt every message. Alternatively, a prearranged prefix could be attached to each message so that the recipient need only decrypt messages with recognized prefixes. To keep even the multiplicity of a prefix's use from being revealed, a different prefix might be used each time. New prefixes could be agreed upon in advance.
  Authentication is also quite useful in systems without identification. Public-key pseudonyms can be used in the systems. Only the untraceable owner of such a pseudonym would be able to sign subsequent messages with it. Secure payment protocols have been proposed in which the payer and the payee might be untraceable [12].

- **Disruption**

  A sender's anonymity gives a new problem: participants can compromise the network by sending an abundance of messages to all or individual participants. This is called disruption. The interesting part of the problem is finding disrupters without reducing the untraceability of other senders. A theoretical solution appears in [10], but it is quadratic in the number of participants, and thus infeasible.

In [25] an efficient, practical solution is presented. The idea is to adjust the well-known ALOHA [49] system to this case. It works very well in practice in the case of a low average load. The idea is that a participant sends whenever she has something to send. If the average load is low, this will not cause a collision in most cases, so there is no delay. If there is a collision, she must try again after some time. If the amount of sending is limited, the disrupters can be caught with opening method.

- **Key-Sharing Graphs**

  In large systems it may be desirable to use fewer than the m(m-1)/2 keys required by a complete graph. If the graph is merely a cycle, then individuals acting alone learn nothing, but any two colluders can partition the graph, perhaps fully compromising a participant immediately between them.
  A different topology assumes the existence of a subset of participants wherein each participant is sufficiently unlikely to collude, such as participants with conflicting interests. This subset constitutes a fully connected subgraph, and the other participants each share a key with every member. Every participant is then untraceable among all the others, unless all members of the completely connected subset cooperate.
  If many people wish to participate in an untraceable communication system, hierarchical arrangements may offer further economy of keys.

- **Underlying Communication Techniques**

  A variety of underlying communication networks can be used, and their topology need not be related to that of the key-sharing graph.

  Efficient use of many other practical communication techniques requires participants to group output bits into blocks. For example, in high-capacity broadcast systems, such as those based on coaxial cable, surface radio, or satellites, more efficient use of channel capacity is obtained by grouping a participant's contribution into a block about the size of a single message.

5.2 MIX Network

Based on the above analyses of the MIX network threats for privacy, some issues related to MIX network design are discussed in [28].

- **Anonymity Versus Pseudonymity**
  Probably the most important design issue is that of anonymity versus pseudonymity. By pseudonymous, we mean that some nodes know the user's pseudonym but they cannot link a pseudonym with a real-world identity. Another option is to have the user be anonymous in the MIX network but be pseudonymous in its dealings with other users (half-pseudonymity). The most important advantages of both anonymity and pseudonymity are as follows.

  Anonymity: Provides better security since if a pseudonym is linked with a user, all future uses of the pseudonym can be linked to the user.

  Pseudonymity:
  1. It is best for privacy protection and accountability. Since pseudonyms have a persistent nature, long term relationships and trust can be cultivated.

2. Pseudonym-based business models (for MIX node operators) are more attractive than anonymity based ones.
3. Abuse control is easier to deal with when pseudonyms are used.
4. Authentication is easier: either Brands credentials [42] or Chaumian blinding [13] can be used when using anonymity.
5. Allows non-interactive processes.

- **Routing**
  For large Internet-based systems especially, having the user choose the nodes in his route randomly doesn't seem like a viable option because:
  1. The nodes and users must know each other. This might be impractical.
  2. Some servers are far from each other and it doesn't make sense from a performance viewpoint to have, for example, a route consisting of nodes in North America, China and Europe.
  3. Nodes should be socially independent. Ideally, the nodes in a route should belong to different organizations and be located in different legal jurisdiction. The whole idea behind using more than one node is that none of them have enough information to determine sender-recipient matchings. Hence, if all nodes in a route belong to the same organization we might as well just use a single node. The motivation for having nodes in different legal jurisdiction is that more than one subpoena needs to be obtained to compromise communications legally.

  Creating good network topologies and route finding algorithms with respect to security and efficiency is not trivial.

- **Dummy Messages**
  Dummy traffic is often used in an unstructured manner and so might not be as effective as it could be. Note the following observations.
  1. If a node sends its message to less than t' nodes, dummy messages are sent in such a way that t' nodes receive messages. The larger t', the harder it is to mount the brute search attacks.
  2. Each node should send messages to at least t'' destinations outside the MIX network (dummy messages should be used to fill the gaps). The larger t'', the harder it is to mount the brute search attacks. Furthermore, this technique also seems to complicate attacks in which the adversary monitors the exit nodes.
  3. In order to randomize the user's communication patterns, having the user send dummy traffic to the entry node should be considered. The challenge is to have good security and minimize the amount of dummy messages used.
  4. Dummy messages could also be used to reduce the amount of time messages stay at a given node. It seems that waiting for c messages to enter a MIX node before sending b $(b>c)$ has similar security properties as waiting to receive b messages before releasing them. This trick could be used to reduce the time messages wait at nodes.

- **Node Flushing Algorithm**
  As seen in Section 3.4, there are many different approaches to flushing nodes. There is a security and practicality tradeoff: in most settings, the longer messages can stay in MIX nodes, the better the security.

- **Packet Size**
  In many situations, using different message sizes yields substantial performance improvements. For example TCP connections require on average one small control packet for every two (large) data packet. It might be inefficient for small messages to be padded or large packets split up in order to get a message of the correct size. As usual in cryptography, there is a security/performance tradeoff: Using more than one message size gives better performance

but worse traceability. We strongly suspect however that there may be other techniques that improve the security properties of the multiple packet size option.

- **Privacy Protection**
  In many situations, the user needs to retrieve some information from a query server, for example, network configuration information, pseudonym public keys, etc. These queries shouldn't erode privacy: the query servers shouldn't obtain non-trivial information about sender-recipient matches. The obvious approach to this problem is to have the user download the entire database but unfortunately, the amount of data to transfer might be too large. We suspect that private information retrieval protocols might be very useful in these situations.

5.3 Onion Routing

Onion Routing network design has the same issues with MIX network design in some ways, e.g., dummy messages, routing, node flushing algorithm, anonymity and pseudonymity. In addition, some implementation issues of Onion Routing network are discussed as follows.

In order to make the Onion Routing effective, there must be significant use of all the nodes, and Proxy Nodes must also be intermediate routing nodes.

For more security, network traffic must be relatively constant. This requires sending dummy traffic over a connection when traffic is light and buffering data when traffic is heavy.

Any period of inactivity on the out-bound queues is likely to be followed by a sequence of onion cells being output on a single queue. Such an implementation makes tracking easier and should be avoided.

In a multi-threaded implementation, there is a significant lure to rely upon the apparent scheduling randomness to reorder events. If reordering is important to the secure operation of the system, deliberate reordering is crucial, since low level system randomness may in fact be predictable.

6. Conclusion

This document reviews the many of the solutions that have been proposed and implemented for network privacy. We have also analyzed these approaches based upon the threats from different types of attacks.

While exposures exist in these approaches, it is clear that:

(1) Some of the attacks require significant resources to launch and maintain. This means that for the most part, these approaches are reasonably secure. As such one or several of the previous developments in this area may be used to provide network-level anonymity.
(2) There appears to be some research opportunities in the development of improved privacy networks.

This work was instigated through our involvement in the Privacy Incorporated Sofware Agent (PISA), European 5th Framework project. We will do further research and development of a model for an **Anonymous Internet Infrastructure (AII),** based on gents which incorporate privacy enhancing technologies**.**
At this point key developments that will be of particular importance for privacy and electronic commerce include the following:

- ⇨ Secure distributed logs. To provide privacy accountability, some way of tracing and proving the way in which data has been handled among disparate partners is needed. This requirement stems from the important requirement in law for accounting practices in the management of private data. It contrasts with the need for anonymity wherein location, timing and other details of interactions are hidden. Irrespective of the anonymity approach taken for our PET solutions, this appears to be an opportunity for our research.
- ⇨ Companies like Zero Knowledge Systems offer anonymity and pseudonymity as a service. These approaches may be used to provide a network layer approach for addressing some of the privacy issues for online commerce. In terms of Agent applications, the anonymizer service must be capable of supporting the communication infrastructure for the agent environment.
- ⇨ Alternate anonymity approaches. There may be better alternatives to provide network anonymity. For instance, an area that has not been explored in this work for network privacy and software agents is peer-to-peer networking. There have been developments over the past several years of technologies to exploit peer-to-peer communication. Some of these approaches purport to offer user anonymity as well as message integrity and secure communication.

Acknowledgments

We would like to thank all members of IIT at the NRC of Canada for their support towards our R&D project on Network Privacy Protection. We also thank our European Community partners in the Privacy Incorporated Software Agent (PISA), EU Fifth Framework Project.

References

[1] A.back, I.Goldberg and A.Shostack. Freedom 2.1 Security Issues and Analysis. May 2001. Available at http://www.freedom.net/info/whitepapers/Freedom_Security2-1.pdf.

[2] A.Back. Hashcash. Available at http://www.cypherspace.org/~adam/hashcash/. March 1997.

[3] A.Juels and J.Brainard. Client Puzzles: A Cryptographic Defence against Connection Depletion Attacks. In S.Kent, editor, NDSS '99 (Networks and Distributed Security Systems), pages 151-165, 2000.

[4] A.Lysyanskaya, R.Rivest and A.Sahai. Pseudonym Systems. Selected Areas in Cryptography : 6th Annual International Workshop, SAC'99, Volume 1758 of Lecture Notes in Computer Science, pages 184-200, Springer-Verlag, 1999.

[5] The Anonymizer. Available at http://www.anonymizer.com.

[6] A.Pfitzmann, B.Pfitzmann and M.Waidner. ISDN-MIXes - Untraceable Communication with Very Small Bandwidth Overhead. Proc.Kommunikation in verteilten Systemen, IFB 267, pages 451-463, Springer-Verlag, 1991.

[7] A.Pfitzmann and M.Waidner. Network without User Observability. Computers & Security, vol.2, no.6, pages 158-166, 1987.

[8] A.Pfitzmann and M.Waidner. Networks without User Observability - Design Options. In Advances in Cryptology - Eurocrypt '85, Volume 219 of Lecture Notes in Computer Science, Springer-Verlag, 1985.

[9] C.Dwork and M.Naor. Pricing via Processing or Combating Junk Mail. In Ernest F.Brickell, editor, Advances in Cryptology - CRYPTO '92, Volume 740 of Lecture Notes in Computer Science, pages 139-147, Springer-Verlag, 1992.

[10] D.Chaum. The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. Journal of Cryptology, vol.1, no.1, pages 65-75, 1988.

[11] D.Chaum and J.Evertse. A secure and privacy-protecting protocol for transmitting personal information between organizations. In Advances in Cryptology—CRYPTO '86, pages 118-167, Springer-Verlag, 1986.

[12] D.Chaum. Security Without Identification: Transaction Systems to Make Big Brother Obsolete. Communication of the ACM, vol.28, no.10, pages 1030-1044, October 1985.

[13] D.Chaum. Blind signatures for untraceable payments. In R.L.Rivest, A.Sherman, and D.Chaum, editors, Proc.CRYPTO '82, pages 199-203, New York, 1983.

[14] D.Chaum. Untraceable Electronic Mail, Return Address, and Digital Pseudonyms. Communications of the ACM, vol.24 no.2, pages 84-88, 1981.

[15] Daemon9. Project neptune. Phrack Magazine, 48(7): File 13 of 18, 8 November 1996. Available at http://www.fc.net/phrack/files/p48/p48-13.html.

[16] D.Goldschlag, M.Reed and P.Syverson. Onion Routing for Anonymous and Private Internet Connections. Communication of the ACM, vol.42, no.2, pages 39-41, 1999.

[17] D.Goldschlag, M.Reed and P.Syverson. Hiding Routing Information. In R.Anderson, editor, Information Hiding: First International Workshop, Volume 1174 of Lecture Notes in Computer Science, pages 137-150, Springer-Verlag, 1996.

[18] D.Kesdogan, J.Egner and R.Buschkes. Stop-and-go MIXes Providing Probabilistic Security in an Open System. In David Aucsmith, editor, Information Hiding: Second International Workshop, Volume 1525 of Lecture Notes in Computer Science, pages 83-98, Springer-Verlag, 1998.

[19] D.R.Simon. Anonymous Communication and Anonymous Cash. In Advances in Cryptology – CRYPTO '96, Volume 1109 of Lecture Notes in Computer Science, pages 61-73, Springer-Verlag, 1996.

[20] Private Credentials. Zero-Knowledge Systems, Inc. white paper, 2000. Available at http://www.freedom.net/info/whitepapers/credsnew.pdf.

[21] R.Samuels and E.Hawco. Untraceable Nym Creation on the Freedom 2.0 Network. Zero-Knowledge Systems, Inc. white paper, 2000. Available at http://www.freedom.net/info/whitepapers/Freedom-NymCreation.pdf.

[22] G.J.Simmons. The history of subliminal channels. IEEE Journal on Selected Area in Communications, vol. 16, no.4, pages 452-462, May 1998.

[23] I.Goldberg and A.Shostack. Freedom Network Whitepapers.

[24] I.Goldberg, D.Wagner and E.Brewer. Privacy-Enhancing Technologies for the Internet. In *Proceedings of IEEE COMPCON '97*, pages 103-109, 1997.

[25] J.Bos. Detection of Disrupters in the DC Protocol. In Advances in Cryptology - Eurocrypt '89, Volume 434 of Lecture Notes in Computer Science, pages 320-327, Springer-Verlag, 1989.

[26] J.Borking. Proposal for Building a Privacy Guardian for the Electronic Age. In H.Federrath, editor, Anonymity 2000, Volume 2009 of Lecture Notes in Computer Science, pages 130-140, Springer-Verlag, 2000.

[27] John Kelsey. Private Communication, 1999.

[28] J.Raymond. Traffic Analysis: Protocols, Attacks, Design Issues, and Open Problems. In H.Federrath, editor, Anonymity 2000, Volume 2009 of Lecture Notes in Computer Science, pages 10-29, Springer-Verlag, 2000.

[29] L.Chen. Access with pseudonyms. In Ed Dawson and Jovan Golic, editors, Cryptography: Policy and Algorithms, Volume 1029 of Lecture Notes in Computer Science, pages 232-243, Springer-Verlag, 1995.

[30] L.Cottrell. Mixmaster. Available at http://www.obscura.com/~loki/.

[31] L.Lamport, R.Shostak and M.Pease. The Byzantine Generals Problem. ACM TOPLAS, vol.4, no.3, pages 382-401, 1982.

[32] M.Pease, R.Shostak and L.Lamport. Reaching Agreement in the Presence of Faults. JACM, vol.27, no.2, pages 228-234, 1980.

[33] M.Reed, P.Syverson and D.Goldschlag. Anonymous Connections and Onion Routing. IEEE Journal on Selected Areas in Communications, vol.16, no.4, pages 482-494, May 1998.

[34] M.Reiter and A.Rubin. Anonymous Web Transactions with Crowds. Communications of the ACM, vol.42, no.2, pages 32-48, 1999.

[35] M.Reiter and A.Rubin. Crowds: Anonymity for Web Transactions. ACM Transactions on Information and System Security, vol.1, pages 66-92, 1998.

[36] M.Waidner. Unconditional Sender and Recipient Untraceability in Spite of Active Attacks. In Advances in Cryptology - Eurocrypt '89, Volume 434 of Lecture Notes in Computer Science, pages 302-319, Springer-Verlag, 1989.

[37] Mature NymIP Network: IP-Layer Desiderata. V1.1, October 2000. Available at http://nymip.velvet.com/cvs/general/zks-desiderata.html.

[38] O.Berthold, H.Federrath and S.Kopsell. Web MIXes: A System for Anonymous and Unobservable Internet Access. In H.Federrath, editor, Anonymity 2000, Volume 2009 of Lecture Notes in Computer Science, pages 115-129, Springer-Verlag, 2000.

[39] P.Boucher, A.Shostack and I.Goldberg. Freedom Systems 2.0 Archtecture. December 2000. Available at http://www.freedom.net/info/whitepapers/Freedom_System_2_Architecture.pdf.

[40] P.Syverson, M.Reed and D.Goldschlag. Onion Routing Access Configurations. In DISCEX 2000: Proceedings of the DARPA Information Survivability Conference and Exposition, Hilton Head, SC, IEEE CS Press, pages 34-40, January 2000.

[41] R.Hes and J.Borking. Privacy-Enhancing Technologies: The Path to Anonymity. Revised Edition. A&V-11. Den Haag: Registratiekamer, 1998.

[42] S.A.Brands. Restrictive blinding of secret-key certificates. Technical Report CS-R9509, CWI-Certrum voor Wiskunde en Informatica, February 28, 1995.

[43] S.Dolev and R.Ostrovsty. Efficient Anonymous Multicast and Reception. In Walter Fumy, editor, Advances in Cryptology - EUROCRYPT '97, Volume 1233 of Lecture Notes in Computer Science, pages 395-409, Springer-Verlag, 1997.

[44] W.Dai. Pipenet 1.1. Available at http://www.eskimo.com/~weidai/pipenet.txt, 2000.

[45] W.Dai. Private Communication, 1999.

[46] N.Zhang, Q.Shi and M.Merabti. Anonymous Public-key Certificates for Anonymous and Fair Document Exchange. IEE Proceedings Communication, vol.147, no.6, pages 345-350, December 2000.

[47] K.Oishi, M.Mambo and E.Okamoto. Anonymous Public Key Certificates and Their Applications. IEICE Trans. Fundam. Electron. Commun. Comput. Sci., E81-A, (1), pages 56-64, 1998.

[48] F.Bao, R.Deng and W.Mao. An efficient and Practical Fair Exchange Protocols with Off-line TTP. Proceedings of IEEE symposium on security and privacy, Oakland, California, USA, pages 77-85, May 1998.

[49] D.Davies, D.Barber, W.Price and C.Solomides. Computer Networks and their Protocols. John Wiley and Sons, 1979.

[50] O.Berthold, H.Federrath and M.Kohntopp. Project "Anonymity and Unobservability in Internet". Available at http://www.inf.tu-dresden.de/~hf2/publ/2000/BeFK2000cfp2000/.

[51] I.Goldberg. A Pseudonymous Communications Infrastructure for the Internet. Ph.D. thesis, University of California at Berkeley, Fall 2000.

Biography

Larry Korba is the group leader of the Network Computing Group of the National Research Council of Canada in the Institute for Information Technology. He is the leader of the Canadian contribution to the Privacy Incorporated Software Agent (PISA) project. His research interests include privacy protection, network security, and computer supported collaborative work.

Ronggong Song received his B.Sc degree in mathematics in 1992, M.Eng degree in computer science in 1996, Ph.D. in network security from Beijing University of Posts and Telecommunications in 1999. He had employed as Network Planning Engineer at Telecommunication Planning Research Institute of MII, P.R.China, and Postdoctoral Fellow at University of Ottawa, Canada. Now, he is working at NRC of Canada. His research interests are privacy protection, network security, e-commerce, IP mobility and QoS.