uOttawa

L'Université canadienne
Canada's university

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Christopher Desmarais
AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.A. (Computer Science)
GRADE / DEGREE

School of Information Technology and Engineering
FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Pluto: A Protocol for Privacy Management In Electronic Communities

TITRE DE LA THÈSE / TITLE OF THESIS

Shervin Shirmohammadi
DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Nicolas Georganas
CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Peter Liu

Ali Miri

Gary W. Slater
Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Pluto: A Protocol for Privacy Management In Electronic Communities

by
Chris Desmarais

A thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements for the degree of

Master of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Information Technology and Engineering

Faculty of Engineering
University of Ottawa

© Chris Desmarais, Ottawa, Canada, 2008

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-41623-5
Our file Notre référence
ISBN: 978-0-494-41623-5

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Abstract

As computers become increasingly connected it is important to ensure that both legal and ethical guidelines are built into the network infrastructure. One key and often underdeveloped area is the management of private (or personal) information. In order for electronic communities to share privacy sensitive information it is important to understand the legal issues surrounding privacy and to design community tools to ensure that there is no unmanaged release of private information. The proposed Pluto protocol has been designed to work with existing privacy legislation to create a solution to managing privacy information for electronic communities; specifically the Pluto protocol was designed as a solution to the Canadian privacy guidelines detailed in "The Personal Information Protection and Electronic Documents Act". This thesis examines existing privacy legislation, explains the proposed Pluto protocol solution and concludes by examining performance evaluation through a proof-of-concept prototype

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Acknowledgements

I would like to express my sincere gratitude to my thesis supervisors Dr. Shervin Shirmohammadi and Dr. Nicolas D. Georganas, for giving me an opportunity to pursue my graduate studies and providing valuable support and guidance. For the last few years you have been a source of encouragement and enthusiasm; your tutelage has pulled from me a better and more versatile solution than I could have originally imagined.

I would also like to thank my colleagues at the Discover Lab, University of Ottawa, for providing me a pleasant working environment. In particular, I would like to thank Dr. Xiaojun Shen for his advice, assistance and leadership in co-writing and submitting the published papers that have come from this research.

Alex Cameron provided considerable insight into privacy issues during the early stages of this research, and his assistance was instrumental in developing this solution. Thank you.

Editing support of this paper was provided by John Desmarais and Jesse Scoble. Thank you for helping me to express my ideas.

Finally, thank to all my wife, Karen Desmarais, and family, for giving me endless love, care and support.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Table of Contents

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

List of Figures

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

List of Tables

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

List of Definitions

Agent	A tool that a user might have to accomplish some goal.
Autonomous	Autonomous software is designed to work without human intervention
Authentication	To prove one's identity. Often refers to a protocol for proving one's identity.
Centralized Solution	A solution, in a distributed environment, that has a critical component "centralized" on one machine in the network.
Child	The target of a relationship. A parent has a relationship to a child.
Client	A user of a system.
Communication Channel	A medium that can be used for passing messages. Often
Distributed Solution	A solution, in a distributed environment, that does not require any single machine to function.
Electronic Community	A collection of people and services that interact through electronic medium.
Encryption Algorithm	A method of securing data so that it is only useable by the intended recipient.
Kerberos	An authentication protocol.
Parent	The source of a relationship. A parent has a relationship to a child.
Purpose	A justification. In this context, a purpose is an explanation of why private data might be stored, transmitted or used.
P3P	Platform for Privacy Preferences, an privacy control protocol for the narrow purpose of passing data from a client to a server.
Privacy Control Component	In the Pluto protocol, a Privacy Control Server that creates Pluto session tickets.
Privacy Control Server	Any service that controls private information flow in a community.
Private Key Encryption	An encryption algorithm that can only be encrypted or decrypted when a shared private key is known.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Privacy	The legal and ethical principle that individuals control data that can identify or be identified with themselves.
Privacy Map	A graph (or network) that models information about "reasonable expectation" relationships between purposes.
Public Key Encryption	An encryption algorithm that can be encrypted by any number of people who have the public key, but only decrypted by one who posseses a special private key.
Resource	Any entity with limited availability. In the context of electronic communities, this will often be a special purpose server or collection of servers.
Role Based Access Control (RBAC)	A method for restricting access to authorized users that bundles permissions into roles and associates users to those roles.
Server (Data Server)	An application in any client-server architecture. In this context, Data Servers are an instance of server that hold and might be able to release private information.
Ticket	An electronic certificate that can be used to prove that a client has certain permissions.
Time Expiry Mechanism	A way to date an a file (or ticket) so that it will be easy to determine if the file is still valid, or if it should be discarded because it has lived past its intended lifespan. Typically, this is done by adding a timestamp indicating the time the file was created and adding Time To Live information that indicates how long the file's expected lifespan is.
Time To Live	The intended lifespan of a file (or ticket).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Chapter 1

Introduction

1.1 Motivation

Software systems are no longer restricted to a single machine (or for that matter even to any simple network). The internet, and in particular web services hosted on the internet, and other networks are continually expanding the reach of numerous applications across machine and network boundaries. Electronic communities where applications are spread across a broad network to share resources and data are becoming increasingly common. As an interesting example of an electronic community is the online insurance business. Not only is there a large number of individual insurance companies available on the internet, but there are also electronic brokers such as www.geico.com that are able to automate comparison shopping by treating competing companies as single community.

At the same time, some of community resources should never be available or available only under restricted conditions. In particular, information that is privacy sensitive (credit card information for example) should never be available to everybody in a network; but only to certain users in very controlled scenarios.

Obviously, there is a tension between these two forces: the *growth* of formal and informal electronic communities, and the *need* to be very selective about how resources and information are shared in that community. The questions that arise from the consideration of these competing forces are:

- can private information **ever** be made available or stored in an electronic community? And if so,
- how can it be controlled in an informal electronic community where applications vary and users have no prior knowledge of each other?

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

1.1.1 Motivating Examples

Three motivating examples, which will be discussed periodically throughout this thesis when they might help illuminate an issue, are described below:

1.1.1.1 E-Commerce Example

A group of web vendors would like to share common customer information such that they can enhance their customers' online experience. The idea is that by sharing their customers' shopping history and browser usage logs they can personalize the online shopping experience across a set of participating vendors creating a virtual shopping mall for customers.

1.1.1.2 Medical Example

In order to make informed decisions a doctor would like to be able to access a patient's entire medical history. The problem is that a patient may have visited several doctors in his/her lifetime, so the patient's history will be scattered across several offices and databases. Doctors would like to be able to share patient records with other doctors.

1.1.1.3 Security and Policing Example

Criminals and criminal activities do not generally respect borders; and so it is quite likely that the police reports related to a crime may be scattered across cities, provinces or even countries. In order to solve inter-jurisdictional crimes, it is often necessary for police departments from different jurisdictions to share police reports across a shared network.

1.2 Research Problem

An electronic community is defined as a community of people and organizations that correspond electronically, generally across a computer network (the work presented here

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

is in fact designed for computer networks but might be adapted for other networks). Typically, the members of an electronic community are in a constant state of flux as new members join or leave the community. Because of the constant change in membership, it is difficult for every member to develop relationships with every other community member. Electronic communities will often develop services that enhance the ability of the community to control communication.

Privacy in the ideal state requires that specific information (generally pertaining to a specific individual or group) needs to be protected such that inappropriate usage and sharing cannot occur. Many countries have legal frameworks and guidelines for the proper use of data that can be traced to a specific person (or group of people). In undertaking this research, Canadian privacy legislation has been used and the expectation is that the research should be equally valid in most countries with similar laws.

The central focus of this research is to enumerate the requirements for sharing private information in an electronic community, and to propose a protocol that addresses those requirements.

1.3 Research Objective

This research is motivated by the following real world problem:

- how can privacy be ensured in a dynamic network of applications and clients?

Privacy requires very strict control of:

- who can access information and
- under what circumstances they can access it.

Considering these requirements, the dynamic nature of most networks makes it difficult to maintain complete control of network information to enable exacting protocols to be applied In order to create a privacy preserving solution, the first part of the research was directed at gaining a complete understanding of existing Canadian privacy legislation. This understanding could then be used to design the technical protocol that could control

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

the sharing of private information across a large dynamic network.

Examination of the privacy legislation identified three critical (and several minor) concerns which are presented below:

- An absolute requirement of a privacy preserving network is the secure transmission of data. Because the users in most networks are not easily controlled, it is possible (and almost certainly expected) that malignant agents will find a way into the system. If malignant users can steal private information as it crosses the network or traverses between networks, then privacy cannot be preserved.

- A privacy preserving network has to authenticate both:
  - the data recipient and
  - the reason for requesting the data.

  Much of the legislation deals with the concept of purpose and sets out the following parameters to define the data system:
  - data must be stored with the purpose for which they can be used,
  - data requests and usage must have a purpose,
  - data release should only occur if the purpose for which the data are being stored matches the reason for which they are being used.

  In order to preserve privacy using a protocol, such as Pluto, it is essential to use the legal parameters of purpose as the context under which data sharing can occur.

- A privacy preserving network must be managed by an accountable agent. The legal requirement is:
  - if private data are released then they must have been released because a specific officer authorized the release.

  Any privacy preserving network be designed so that it can be controlled by a person. That person will require a tool (or set of tools) able to manage private information in the system.

The research objective for this thesis is to understand the legal and ethical issues with the design of a privacy preserving network and to use that understanding to design a protocol

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

for electronic communities that would enable the controlled sharing of private information.

1.4 Research Contributions

This thesis contains a number of contributions, including:

- A detailed analysis of Canadian Privacy Standards, leading to system requirements, examining important considerations for sharing private information across a network.
- Analysis of related work, such as Kerberos and P3P, with discussions of the reasons for their failure or their need for extension as a general solution for privacy control.
- Design of a novel legislation-compliant, purpose based protocol for Privacy Control for dynamic electronic communities.
- Implementation and performance evaluation of a prototype as proof of concept and validation of the design and theory.
- Peer reviewed publications:
    - Submitted: C. Desmarais, X. Shen, S. Shirmohammadi, A. Cameron, N.D. Georganas, I. Kerr, "Pluto - A Privacy Control Protocol for e-Commerce Communities", J. Electronic Commerce Research.
    - C. Desmarais, X. Shen, S. Shirmohammadi, A. Cameron, N.D. Georganas, I. Kerr, "PLUTO - A Privacy Control Protocol for e-Commerce Communities", Proc. IEEE Conference on Enterprise Computing, E-Commerce and E-Services, Tokyo, Japan, July 2007.
    - C. Desmarais, X. Shen, S. Shirmohammadi, A. Cameron, N.D. Georganas, I. Kerr, "Privacy Management in e-Commerce Communities", Proc. IEEE Canadian Conference on Electrical and Computer Engineering, Vancouver, BC, Canada, April 2007.
    - X. Shen, S. Shirmohammadi, C. Desmarais, N.D. Georganas, I. Kerr, "Enhancing e-Commerce with Intelligent Agents in Collaborative e-Communities", Proc. IEEE Conference on Enterprise Computing, E-

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Commerce and E-Services, San Francisco, CA, U.S.A, June 2006.

- X. Shen, S. Shirmohammadi, C. Desmarais, N.D. Georganas, "Multi-agent System Architecture for Collaborative E-Commerce", Proc. IEEE Canadian Conference on Electrical and Computer Engineering, Ottawa, ON, Canada, May 2006.

1.5 Organization of the Thesis

It should be noted that, due to the multidisciplinary nature of this research which involves both Law and technology, the Related Work section, which is usually chapter 2 in a typical technical thesis, has been moved to chapter 4. The reason is that the application domain's characteristics, namely the Law aspects and legal issues, must be explained first before any meaningful technical discussion of the related work can be done. The thesis is therefore organized as outlined below:

In **Chapter 1 Introduction**, the research motivation has been provided, the problem has been outlined, objective(s) for the thesis stated and the contributions to the research listed.

Chapter 2 Privacy – Standards and Issues provides background information on privacy issues; including a detailed discussion of the Canadian Privacy Standards.

Chapter 3 Requirements Analysis and System Architecture is a discussion of potential network architectures detailing their potential in providing a solution to the stated problem and discussing why they were rejected for this thesis, but also explaining their possible benefits for other applications. This chapter will conclude with a high level description of the proposed Pluto architecture and why it is potentially the best candidate for most networks in providing a solution to the outlined problem.

Chapter 4 Related Work discuss the existing technologies that were adapted as part of the solution; in particular, Kerberos and P3P. The thesis will discuss how these two

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

technologies provide important building blocks for the final solution, and points out why neither can create a general solution on their own.

Chapter 5 Pluto: The Proposed Protocol presents the proposed Pluto protocol and explains how the proposed solution can handle the legal and technical issues pertaining to privacy control.

Chapter 6 Prototype and Performance Evaluation discusses a developed Pluto prototype; and a variety of performance tests that were carried out using the prototype.

Chapter 7 Conclusions and Future Work summarizes and concludes the thesis and provides recommendations for future research.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Chapter 2

Privacy – Standards and Issues

2.1 Introduction

Perhaps the simplest way to summarize privacy concerns is to say that information about a specific identifiable person (or group) is useable only by that person or agents that have been approved by that person, with limited exceptions generally involving policing and national security issues. There is an important legal and ethical difference between *having* information and being able to *use* that information. An organization can have information about a person but is restricted in how they can use it; for example a school can have a student's home address but would not be able to sell that address to credit card companies.

Most countries have legislation that provides for varying levels of privacy protection; privacy protection has even been identified as a basic human right by the United Nations [7].

The law aspects of the work presented in this thesis have been done in consultation with Canadian legal experts, at University of Ottawa's School of Law, using the Personal Information Protection and Electronic Documents Act [5]. This chapter presents the key points in that legislation and discusses mechanics for capturing those points for semi-autonomous protocols for electronic communities.

It is important to state at the outset that the work undertaken as part of this thesis is for a community solution, and not for a specific singular data server. This thesis will make several recommendations for data representation and policies for servers; but there is a disconnect between the communities designed to share data and a server's design to hold data. Where policies would be required to be implemented at the level of a data server, this thesis makes general recommendations but does not attempt to find a full solution. This thesis is concerned with providing a community level solution to the sharing of data

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

in a private environment.

2.2 Personal Information Protection and Electronic Documents Act

The Personal Information Protection and Electronic Documents Act is the basis for the Canadian standard for privacy protection. Passed on April 13, 2000 in response to the increasing ability of technology to use and transmit information of all kinds, the act was designed to provide the legal framework "to govern the collection, use and disclosure of personal information in a manner that recognizes the right of privacy of individuals with respect to their personal information and the need of organizations to collect, use or disclose personal information for purposes that a reasonable person would consider appropriate in the circumstances."[5]

The act was also designed to demonstrate to the European Union that Canada's privacy standards were adequate to protect information concerning European citizens [8]. As such it provides an excellent benchmark for research purposes as it has been shown to be compatible with foreign privacy standards.

Please note that, in the following sections, shaded boxes are extracts from the Personal Information Protection and Electronic Documents Act.

2.3 Accountability

> An organization is responsible for personal information under its control and shall designate an individual or individuals who are accountable for the organization's compliance with the following principles.
>
> Accountability for the organization's compliance with the principles rests with the designated individual(s), even though other individuals within the organization may be responsible for the day-to-day collection and processing of personal information. In addition, other individuals within the organization may be delegated to act on behalf of the designated individual(s).
>
> MODEL CODE FOR THE PROTECTION OF PERSONAL INFORMATION Section [illegible]

The legal principle involved in accountability infers that all decisions made by any

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

organization collecting personal data (and its tools, in particular any software) must be controlled by a designated individual (or small group of people) who is *accountable* for how private information is used. Thus, the designated individual is legally responsible for any private information that is handled improperly by or within the system.

A well designed privacy control system requires privacy management tools that are directly controllable by a person (or small group reporting to that person). This may not be a trivial task, as electronic communities themselves can easily be too complex to be managed by a person, or even a fairly large group.

The general solution, to having a community be accountable to a specific person or office will be to create a simplified set of privacy controlling rules that exist as a community resource for handling personal data as shown in Figure 1. The specific solution proposed in the Pluto architecture is to have a single Privacy Control Component agent in the system that is invoked whenever private or personal information might be involved in a transaction; but it is possible that there are other solutions as will be discussed in Chapter 3.

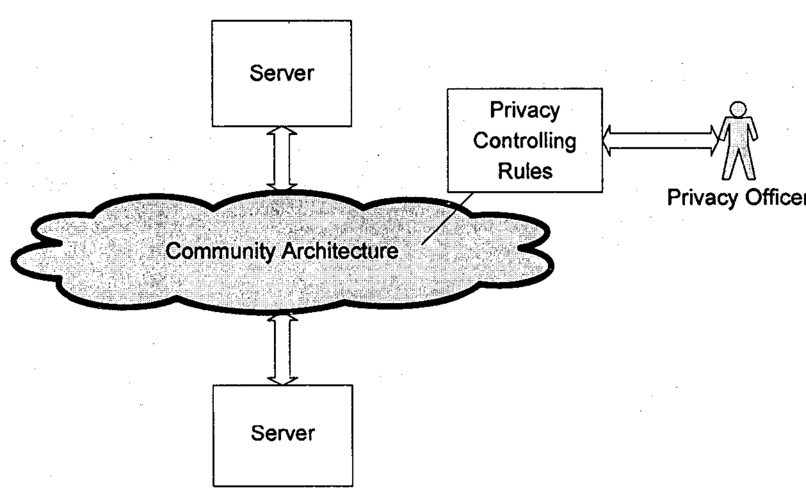

Figure 1 - Requirements for Accountability

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

2.4 Purpose Based Storage

> Identifying the purposes for which personal information is collected at or before the time of collection allows organizations to determine the information they need to collect to fulfil these purposes. The Limiting Collection principle (Clause 4.4) requires an organization to collect only that information necessary for the purposes that have been identified.
>
> The identified purposes should be specified at or before the time of collection to the individual from whom the personal information is collected. Depending upon the way in which the information is collected, this can be done orally or in writing. An application form, for example, may give notice of the purposes.
>
> – MODEL CODE FOR THE PROTECTION OF PERSONAL INFORMATION, Section 4.2

If a person controls all information that relates to that person, then this principle explains how organizations can collect and retain that personal information. The organization can collect and store such information if the person authorizes them to, and more specifically only for the purposes that the person has agreed to. The organization is allowed to act as an agent for the person and hold his information for purposes that the person has agreed to, and it is only for those purposes that the organization is allowed to hold the information.

The impact of this principle is that servers that have private information are required to maintain purpose information that justifies the reason for having that information. Systems might take one of several approaches to this:

- all information in this server is here for a specific purpose; it is implied that all information is stored for the same reason
- each complete record of a person and all the information about that person will be here for a generic purpose; it is implied that all information about a person is stored for the same reason
- each partial piece of information (each data row for example) will have a purpose identified with why that specific piece of information is retained; it is implied that partial information about a person could be stored for a variety of different reasons

From a community point of view though, the specific approach taken by a particular

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

server is not relevant. What is relevant is that all the information on all the servers can be expected to have a reason associated with it, as shown in Figure 2.

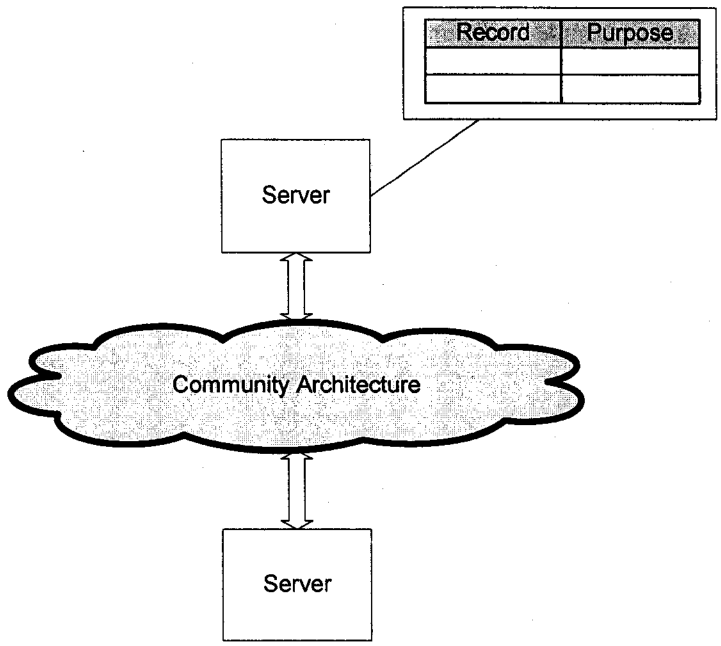

Figure 2 – Requirements for Purpose Based Storage

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

2.5 Consent

> The knowledge and consent of the individual are required for the collection, use, or disclosure of personal information, except where inappropriate.
>
> Note: In certain circumstances personal information can be collected, used, or disclosed without the knowledge and consent of the individual. For example, legal, medical, or security reasons may make it impossible or impractical to seek consent.
>
> In obtaining consent, the reasonable expectations of the individual are also relevant. For example, an individual buying a subscription to a magazine should reasonably expect that the organization, in addition to using the individual's name and address for mailing and billing purposes, would also contact the person to solicit the renewal of the subscription. In this case, the organization can assume that the individual's request constitutes consent for specific purposes. On the other hand, an individual would not reasonably expect that personal information given to a health-care professional would be given to a company selling health-care products, unless consent were obtained. Consent shall not be obtained through deception.
>
> - MODEL CODE FOR THE PROTECTION OF PERSONAL INFORMATION Section 4.3

Consent is a difficult concept for electronic communities; very few privacy controlling communities could reasonably exist if they were required to check with an individual every time his information might be transferred across the network. However, there are two ways that information could be passed through a community:

- if the person consented for his information to be passed to other organizations (often subsidiaries of the first organization)
- passing the private information to others across a community would be part of "reasonable expectations" for another consented usage. If passing the information to another member of the community would be a reasonable expectation of the original consent, then the information should be passable. The examples below, 2.5.1.1 and 2..5.1.2, illustrate how that consent might be implied.

Passing information for directly consented reasons is fairly straightforward (simply ensure that there is consent for passing the information). The concept of reasonable

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

expectation is somewhat more complex but can be considered by implementing a network of purposes that are related by directional "reasonable expectation" relationships. The child purpose of such a relationship is a "reasonable expectation" of the parent purpose. The community can then use this purpose inference network to make decisions about passing information across the network. This is shown in Figure 3.

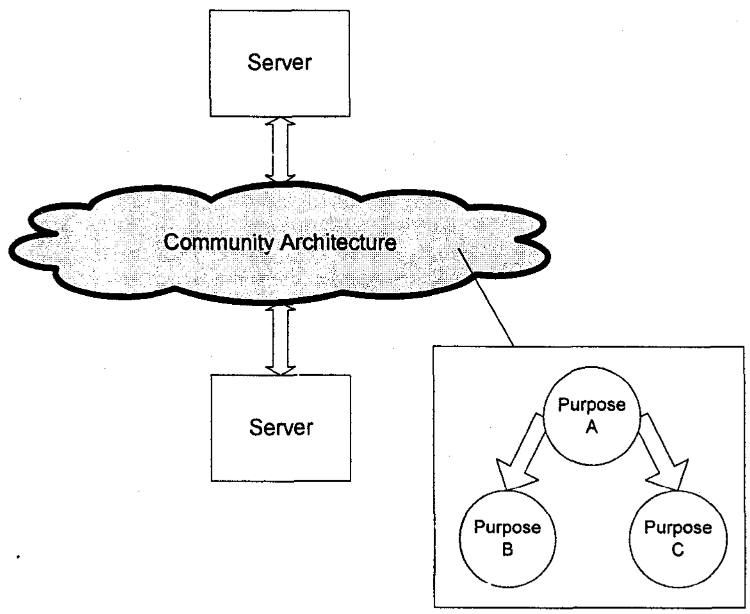

Figure 3 - Requirements for Consent

2.5.1 Motivating Examples

What follows is a brief explanation of how the concept of consent and "reasonable expectation" might allow the passing of private information in the example communities.

2.5.1.1 E-Commerce Example

When a customer purchases a product from one of the participating vendors, that customer is given the option to opt in to the information system network for the purpose of enhancing his shopping experience across all participating vendors. If he elects to opt in then he is consenting to the "enhanced shopping across vendors" purpose. As shown in Figure 4, it is a reasonable expectation that passing private information to other vendors to enhance shopping at their store is a sub-purpose of "enhanced shopping across

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

vendors." Because it is a reasonable expectation, the information could then be passed.

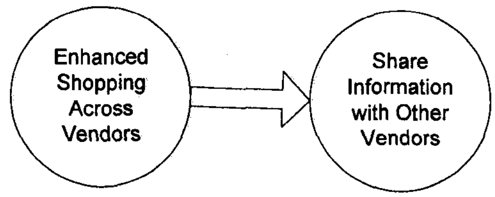

Figure 4 - E-commerce Example of Related Purposes

2.5.1.2 Medical Example

When a patient visits his doctor he will be consenting for the doctor to retain his private information for the purpose of "enhanced medical diagnosis" on future visits. The purpose "sharing medical history with patient's other doctor" is a reasonable expectation of the "enhanced medical diagnosis" purpose, as shown in Figure 5.

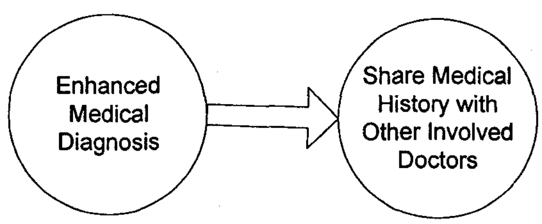

Figure 5 - Medical Example of Related Purposes

2.6 Collection Limitation

> The collection of personal information shall be limited to that which is necessary for the purposes identified by the organization. Information shall be collected by fair and lawful means.
>
> Organizations shall not collect personal information indiscriminately. Both the amount and the type of information collected shall be limited to that which is necessary to fulfil the purposes identified. Organizations shall specify the type of information collected as part of their information handling policies and practices, in accordance with the Openness principle (Clause 4.8).
>
> – MODEL CODE FOR THE PROTECTION OF PERSONAL INFORMATION Section 4.4

The legal principle here is that organizations can only collect information for stated

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

purposes. In Section 2.3 above, it was noted that all private information would need to be stored with purpose information, but there was no discussion of how that information would be collected.

In large part, the initial gathering of information by data servers is done by those servers and is outside the influence of the community. Initial gathering of private information is not something that can be controlled at the level of community; however, the community will need to control how private information is collected by client applications from data servers across the community.

In order for a remote client to collect data from a community data sever, it will need to be collecting that information for a valid purpose. Technically, this will mean that all requests for private information will require embedded purpose information that details why the information is being requested. This will require that all communication between a client and a data server be bound by a purpose that defines the allowed scope of the communication, as shown in Figure 6.

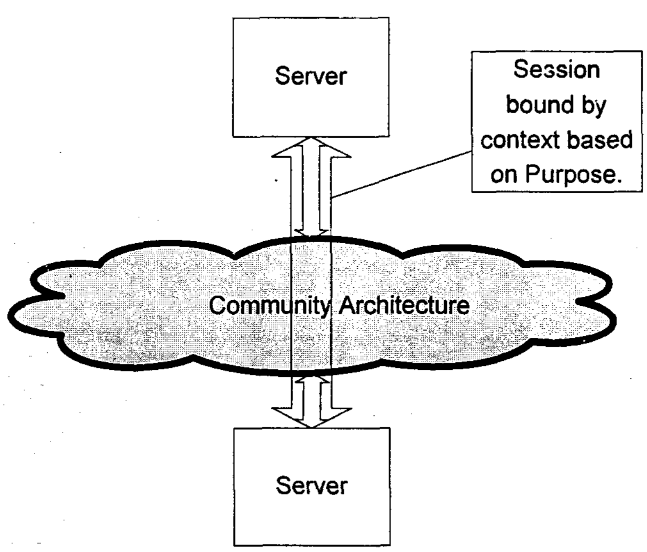

Figure 6 - Requirements for Collection Limitation

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

2.7 Usage Limitation

> Personal information shall not be used or disclosed for purposes other than those for which it was collected, except with the consent of the individual or as required by law. Personal information shall be retained only as long as necessary for the fulfilment of those purposes.
>
> Organizations using personal information for a new purpose shall document this purpose (see Clause 4.2.1).
>
> - MODEL CODE FOR THE PROTECTION OF PERSONAL INFORMATION Section 4.5

As collection and storage require a valid and consented purpose to justify the acquisition and storage of personal information; usage of that information is also limited to the purposes that have been consented to. Privacy violations can occur if the private data is used for purposes that are not related to the original consented purpose.

For the most part. this is beyond the scope of the community infrastructure. The community infrastructure can and should be able to ensure that data is only passed to authorized clients and only for what that client claims his purpose to be. However, once a client has retrieved data the community is no longer in control of what the client does with that information; any violations are the legal responsibility of that client.

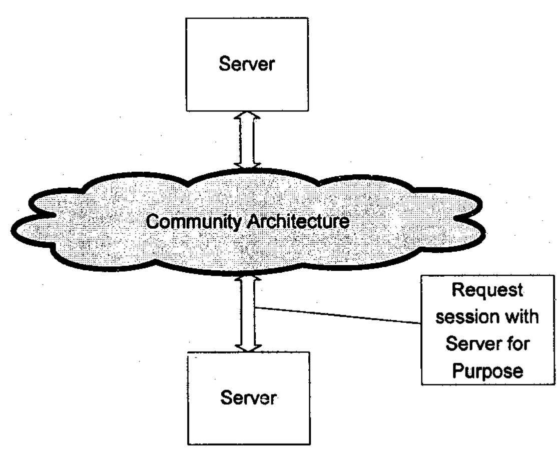

Figure 7 - Requirements for Usage Limitation

What the infrastructure can do however is require that clients include intended purpose limitations in their request for a server connection, as seen in Figure 7. The client will understand that he is expected to limit his usage of the information to those purposes that

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

were part of the original request. The community cannot specifically police what happens in a remote client, but it can ensure that there was an understanding of how the data would be used when it was passed to the client.

2.8 Accuracy

> Personal information shall be as accurate, complete, and up-to-date as is necessary for the purposes for which it is to be used.
>
> The extent to which personal information shall be accurate, complete, and up-to-date will depend upon the use of the information, taking into account the interests of the individual. Information shall be sufficiently accurate, complete, and up-to-date to minimize the possibility that inappropriate information may be used to make a decision about the individual.
>
> - MODEL CODE FOR THE PROTECTION OF PERSONAL INFORMATION Section 4.6

If information about a person might be used, then it should be as up to date as possible. It is not hard to imagine scenarios where a company might make poor decisions based on out of date private information; for example, a bank making a mortgage decision based on a person's credit rating from several years ago.

In many ways, this is not something that can be controlled directly by the community as the community does not directly maintain any data in the system. However, the community itself can be a great resource for assisting the servers to maintain their own databases. The servers could act as clients of other servers in the community and by doing so collect data that might update their own records.

No special recommendations are added to the design of the community to enable the accuracy principle. However, the hope is that by creating an easy to use way to refresh data, clients will be less likely to store data that might go out of date frequently.

2.9 Safeguards

> Personal information shall be protected by security safeguards appropriate to the sensitivity of the information.
>
> The security safeguards shall protect personal information against loss or theft, as well as unauthorized access, disclosure, copying, use, or modification. Organizations shall protect personal information regardless of the format in which it is held.
>
> - MODEL CODE FOR THE PROTECTION OF PERSONAL INFORMATION Section 4

Private information must, of course, be secure. All the controls on purpose and consent

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

are secondary to ensuring that data can't be casually stolen directly from the system. For this reason, authorization and network security play a critical role in any privacy controlled electronic community.

In order to accomplish reasonable levels of security, each client will have to be known at a minimum by the Privacy Officer and be able to prove his identity to that officer in order to use the system. Also, all transmission of private information should be secured by some form of encryption as it passes through the community. These requirements are summarized in Figure 8.

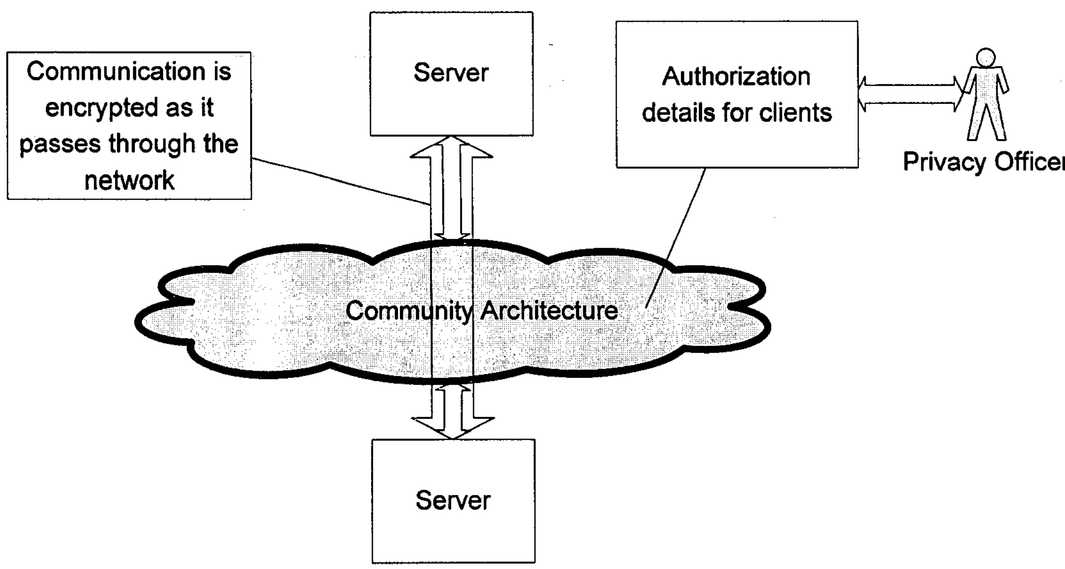

Figure 8 – Requirements for Safeguards

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

2.10 Open Policy

> An organization shall make readily available to individuals specific information about its policies and practices relating to the management of personal information.
>
> Organizations shall be open about their policies and practices with respect to the management of personal information. Individuals shall be able to acquire information about an organization's policies and practices without unreasonable effort. This information shall be made available in a form that is generally understandable.
>
> The information made available shall include
>
> (a) the name or title, and the address, of the person who is accountable for the organization's policies and practices and to whom complaints or inquiries can be forwarded;
>
> (b) the means of gaining access to personal information held by the organization;
>
> (c) a description of the type of personal information held by the organization, including a general account of its use;
>
> (d) a copy of any brochures or other information that explain the organization's policies, standards, or codes; and
>
> (e) what personal information is made available to related organizations (e.g., subsidiaries).
>
> – MODEL CODE FOR THE PROTECTION OF PERSONAL INFORMATION Section 4.8

The Open Policy principle helps define the exact role of the Privacy Officer; in particular, providing details of some of his responsibilities beyond controlling release and usage of private data. The additional duties relate primarily to being able to describe the privacy information and how it is used by the organization.

In particular, two elements of this principle jump out as significant: (d) and (e) require that an organization, in this case the community, be able to describe the policies for usage and transmission of private information to subsidiaries.

Since any community could potentially be described, this principle does not add any new requirements to the system, but it does help to identify a force on the community. The force is for simplicity; the simpler the system the easier it will be for the privacy officer to describe.

This force for simple understandable policies is also present in the accountability principle: if a person will be responsible for private information being passed in an

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

electronic community, then he will need a full understanding of how that is done. This force is also valuable to individuals about whom the system might hold private information; and it has been previously noticed that “the perception of user control over private information is critical to user’s online experience.”[16]

2.11 Disclosure

> Upon request, an individual shall be informed of the existence, use, and disclosure of his or her personal information and shall be given access to that information. An individual shall be able to challenge the accuracy and completeness of the information and have it amended as appropriate.
>
> – MODEL CODE FOR THE PROTECTION OF PERSONAL INFORMATION Section 4.9

The Disclosure principle is the principle that individuals are entitled to access any information an organization might have that is directly related to that individual. Further, an individual is allowed to challenge, correct or complete that information.

The Disclosure principle is complicated because it really is not a community matter. The community does not hold private information; it provides a network to assist in sharing information, but the actual information is held by the servers. Disclosure involves individuals and organizations and does not require interaction with any community architecture. If an individual needs access to personal private information from a data server then the request should be routed directly to the servers.

However, it is possible that the community could be adapted to assist individuals in finding servers that store personal private information and to ask for disclosure of that information. In order to facilitate this, a formal protocol for identifying individuals would need to be developed so that a data server could identify exactly which individual is requesting disclosure. Because this is a sizeable extension and not required of a community this was not fully developed in this thesis, but the extension will be discussed again as potential future work.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

2.12 Challenge

> An individual shall be able to address a challenge concerning compliance with the above principles to the designated individual or individuals accountable for the organization's compliance.
>
> An organization shall investigate all complaints. If a complaint is found to be justified, the organization shall take appropriate measures, including, if necessary, amending its policies and practices.
>
> - MODEL CODE FOR THE PROTECTION OF PERSONAL INFORMATION Section 4.10

The Challenge principle is the principle that individuals can challenge how privacy is handled, and more interestingly that such challenges might require changing the policies and structure of how that information might be used.

Functionally this results in two requirements shown in Figure 9:

- The first requirement is to provide individuals enough information in order to request a change to privacy policies. The Open Policy principle (and to a lesser degree the Disclosure policy) already requires that such information be available, so there is no additional requirements.
- The second requirement is to allow appropriate measures to be taken if that individual's challenge has merit. This requires that the system have rules for dynamically changing the policies of the community. In particular, it must be possible for the purpose map introduced to solve consent related issues to be easily modifiable, and the data server must be able to change the level of involvement with a community (opt out of a community, change or limit the purposes that the community that allow the authorization of connections).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

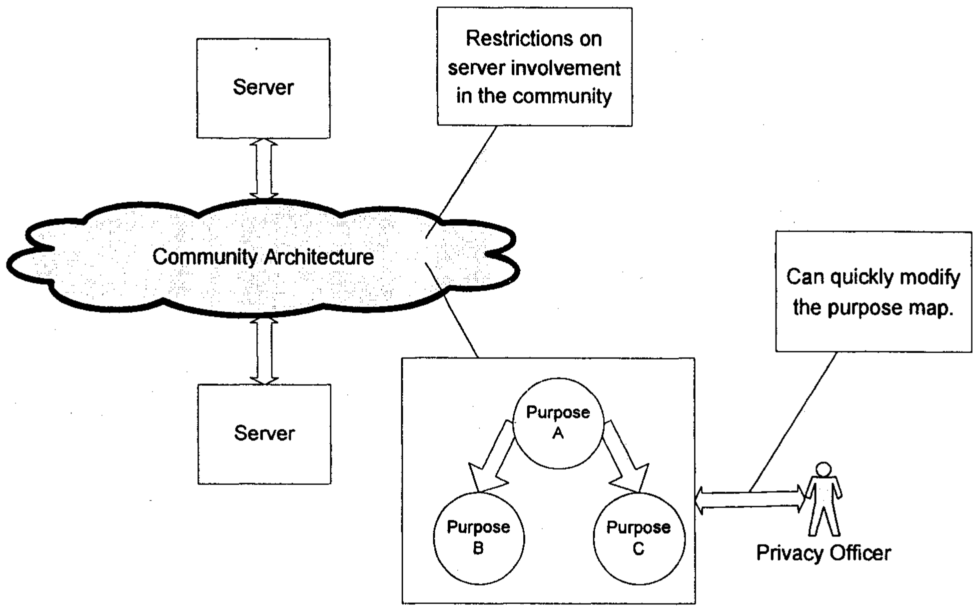

Figure 9 - Requirements for Challenge

2.13 Summary of Legal Requirements

After examining the Canadian Privacy Standards, it is possible to enumerate the requirements for a community that provides the required levels of privacy protection:

- All data are stored with related purpose information that explains how it can be used and shared in the network.
- All requests for data will be scoped by the purpose for which the data will be used.
- All communication between the client and a server will be encrypted to ensure its safe transmission across a network.
- A privacy officer will maintain control over a purpose map that explains how purposes are related
- A privacy officer will maintain details about the authorization of clients to request information in the system, and for which purposes the clients can request information.
- The privacy officer is accountable for private information that is disclosed, and will require tools to support making quick changes to the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

network. This tool, the Privacy Control Component, will control client authorization and the ability to manage the purpose map.

- Servers can restrict the purposes for which they will share information with a community.

These requirements are presented in Figure 10.

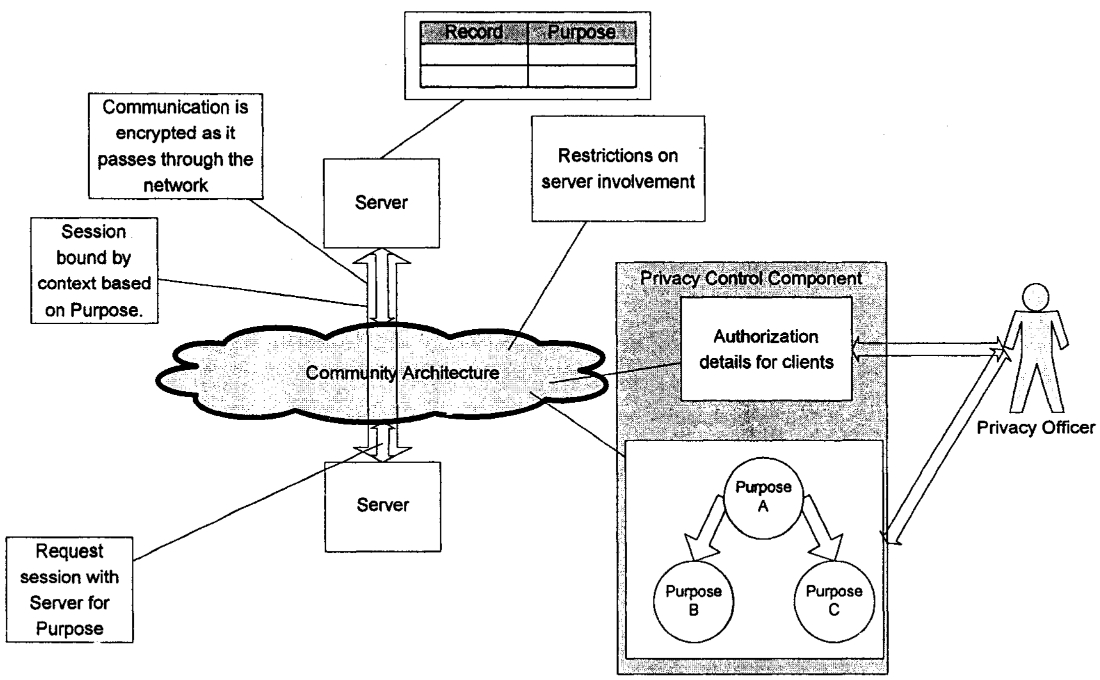

Figure 10 - Requirements Summary

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Chapter 3

Requirements Analysis and System Architecture

3.1 Introduction

Having discussed the requirements for a privacy controlled community, it is now possible to discuss solutions that could meet those requirements. As there are several possible solutions that could meet the requirements, this chapter will start by introducing desireable but not required aspects (forces) of possible solutions that can also be used to judge solutions. Next, some possible high level solutions are presented below along with a discussion of the conditions under which they might make suitable candidate solutions before considering how they might be judged using the non required aspects (forces). The final chosen high level architecture provides the basis for the proposed Pluto protocol: Session Based Control. Although, the recommendation is that Session Based Control is the best general purpose solution; there are specific scenarios where other possible architectures may be superior.

3.2 Forces

The non-required criteria by which possible solutions could be judged are presented in the following paragraphs. These criteria, or forces, have been used to select possible solutions for more detailed examination.

3.2.1 Accountability Support – Responsiveness

One of the requirements examined in chapter 2, was the need for the system to be accountable to a privacy officer. A specific and identifiable person will be legally and morally responsible for the private information passed through the network. As such, any system that is easier for the Privacy Officer to control will be superior to one that is more

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

difficult to control.

One of the important ways in which a system's accountability can be judged is the length of time it takes a Privacy Officer's changes to be acted on by the system. If the Privacy Officer can make changes that immediately affect the system; then the system has good responsiveness. Some of the systems we will be looking at might take several hours to respond to a change by the Privacy Officer; those systems have poor responsiveness.

As the privacy officer is accountable for all actions taken by the system, even if it hasn't immediately followed his directives; any good system will be responsive to the Privacy Officer.

3.2.2 Accountability Support – Absence of Data Duplication

Another important aspect of accountability support is accuracy. While the Privacy Officer should be able to ensure the accuracy of the data on a single server, that job is considerably more complex if the same data is spread across several machines. In particular, some of the following solutions will have the same data on different machines. The problem with duplicating data on different machines is that the data may not stay the same:

- one machine might update its information without telling the other system(s)
- one machine might miss an update from the other machine(s)

Because the Privacy Officer will have difficulty managing data on several machines, any good system will minimize the amount of data duplicated across different servers.

3.2.3 Readability / Simplicity of Privacy Policy

It is important that the policies used by the community be readable, and easily understood, primarily because it is important for legal reasons (see 2.10 Open Policy and 2.12 Challenge). As part of the two legal requirements noted above, it is required that the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

privacy policy information be available to the general public and it is important that the policies used by the community be explainable and easily understood by a layperson.

The Canadian Privacy standard suggests the possibility of a brochure that can be used to explain the privacy guidelines being used by an organization; this seems to be a reasonable guideline for the complexity of information needed to summarize the privacy policy of a community. For that reason, any solutions that might require more policy information than could fit in a brochure are probably inferior to solutions that could be explained in 2-3 pages.

There are also important trust issues; users are unlikely to trust privacy controls that they cannot understand, and more likely to trust privacy that they can understand [16]. With more trust in the system, it is more likely that the system will be able to acquire a person's consent to have their information be useable in the system.

Additionally, simpler more readable privacy policy statements will assist privacy officers in controlling their system. The privacy officer is legally responsible for his system, so the system should be something that this officer can completely understand.

The desirability of readable privacy policies militates against many possible complex machine independent solutions. Any scattering of rules across system agents makes it less likely that the network will be explainable to consenting individuals, and it will complicate the privacy officer's job.

The network needs to be presentable for consent and challenge as well as to simplify the privacy officer's job; so any good network solution will be easily explained and be highly readable.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

3.2.4 Scalability

There are several potential solutions for the various scenarios that would have very small groups of servers:

- One server can be handled with encryption and passwords to enclose requests
- A pair of servers might simply connect via a communication channel so they could easily share data.

However, as the number of servers increases, many simple solutions become unwieldy. For truly large scenarios it is possible that the number of servers will make some solutions unmanageable.

As the intent of this thesis is to design a solution for large dynamic electronic communities, where design decisions cannot be held to a specific number of machines, the work presented here is assumed to be for an unbounded number of servers. Because system performance will always degrade as the number of servers increases; any good system will be minimally impacted as the number of data servers increases.

3.2.5 Network Efficiency

Again, expecting that any proposed solution will need to work for large networks reduces the set of possible solutions. In this scenario; however, the client-server interaction becomes the issue as the computation time to establish controlled communication between clients and servers will add up with increasing number of clients and the number of servers. Each authorized and approved message between a server and a client will take some finite amount of time, so as the number of those messages increases the time it takes to process those messages will increase. As there is a finite amount of time to process messages between clients and servers, any good system will minimize the amount of time required to process those messages.

3.2.6 Bottleneck Avoidance

Because of the need to make the system responsive and accountable to the Privacy Officers and the need to have simplified systems, the best solutions are likely to be

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

centralized solutions where a single system resource (generally a specialized server) manages connections for the entire system. A centralized solution brings new problems; however, since all traffic needs to filter through a single machine, placing an upper limit on the number of messages that can be certified through the system.

As the ideal solution is likely to be one where privacy is controlled by a single machine, that machine will place an upper limit on the amount of communication through the system. Any good system will seek to maximize communication by minimizing the amount of work required by the privacy control server.

3.3 Distributed Control Solutions

A collection of the possible architectures that were considered as possible privacy control solutions are outlined below. The solutions in this section share one common attribute: none rely on a centralized server for privacy control, and instead distribute control across several machines. Ultimately, these solutions were rejected primarily as they were found lacking in accountability controls, readability and/or scaling.

3.3.1 No Collaboration

The most obvious solution is, of course, no community solution as shown in Figure 11. Each data server can manage its own security, authorization and purpose control. Privacy officers maintain unique data for each server. If the expectation is that there will only ever be a single data server, then maintaining this solution is completely reasonable.

The problem is that this scheme tends to break down when there is more than one machine in the network. As the network grows:

- data duplication hurts efficiency and scalability
- accountability is limited to a specific machine, there is no accountability for the network
- a user can only ask for information policies about a specific server

Ultimately, with no system control, there is not any system: just a collection of servers on specific machines. There is no system-wide accountability and there is limited ability to

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

scale.

It seems likely that more sophisticated scenarios will soon emerge in e-commerce, and other domains where personal information might need to be exchanged (medical, security, scholastic etc.). As those more sophisticated scenarios emerge, it seems likely that there will be better solutions.

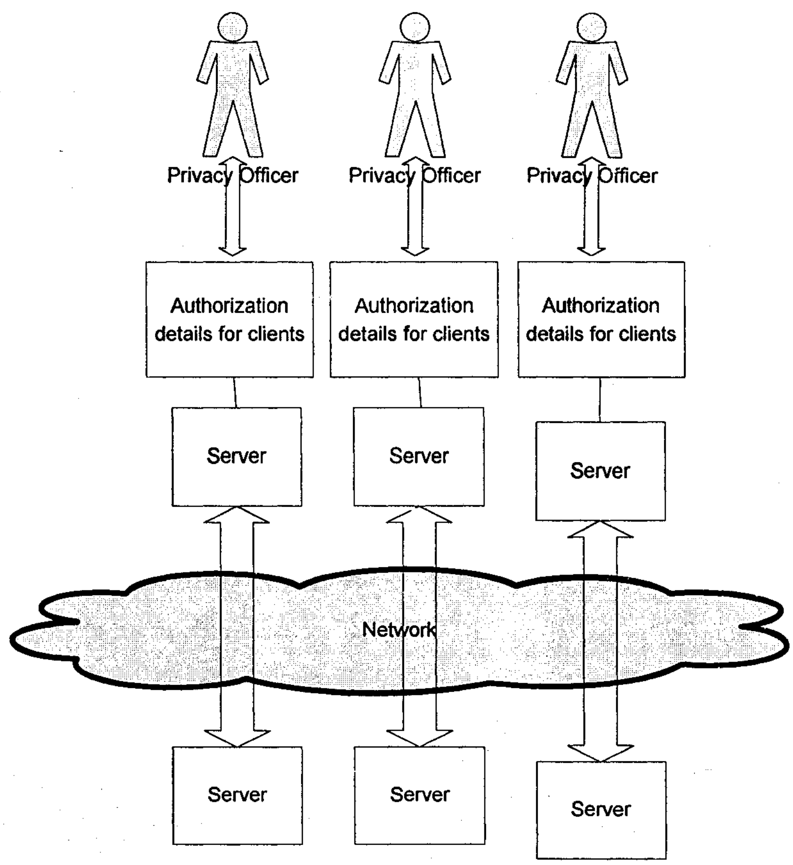

Figure 11 - No Collaboration

3.3.1.1 Accountability Support – Responsiveness

The No Collaboration solution is difficult because although it often appears to be a good solution when considering a single data server, it is often a poor system when considering the entire system. A single data server will be very responsive to the Privacy Officer; changes to a server's data can be made and the server can immediately act on those

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

changes. However, from a community perspective, this solution is not responsive. A privacy officer can immediately change a single server, but changing all the servers will be a much more painstaking process as there is no immediate connection between the servers.

3.3.1.2 Accountability Support – Absence of Data Duplication

Each server maintains its own data with no knowledge of the other servers. With no knowledge of the other servers, it is obviously very difficult to avoid data duplication. Thus, this solution is likely to have considerable data duplication.

3.3.1.3 Readability / Simplicity of Privacy Policy

Again, examining any single server in the network will make this solution seem reasonable. Each machine has a policy that should be easily retrievable and explained. However, the solution still poses problems when dealing with the entire community. Trying to extract information for the entire network is extremely difficult as personal information could be spread across several machines each with their own privacy rules. As a community solution, individual servers do not provided easily retrievable and readable privacy policies.

3.3.1.4 Scalability

As a community solution, scalability is extremely poor. If there are several hundred separate servers that need to be changed to make a change to a policy (e.g. add a purpose), then it will have to be done at each site individually which results in an inefficient system.

3.3.1.5 Network Efficiency

Network efficiency should be quite good; each server makes its own decisions and

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

responses. There should be no extra overhead needed.

3.3.1.6 Bottleneck Avoidance

There is no possible bottleneck because there is no commonly used resource. There may still be problems with server overload or network throughput, but there is no single system resource that will place a limit on system performance.

3.3.2 Inherited Trust

As the number of servers grows, it becomes difficult to justify maintaining unique authorization to access information for each possible client on each server. Duplicating client information is inefficient, and likely to cause errors (as one server updates the client information but another does not). Thus, as network size increases, the next logical step would be to trust peer server to authenticate and authorize clients as shown in Figure 12.

This system could work nicely for a slightly larger number of servers, but would still show problems as the number of servers grew. The largest problem is that no one privacy officer controls the entire system; accountability is lacking because the Privacy Officer for one server must trust a Privacy Officer on another server. If there is a release of improper information, then who is accountable?

Other problems start to happen as the number of connections increases and thus will perform quite badly as the system size increases. With a small network of 3 servers, each server might need to check with 2 other servers: there are six possible cross-check pairs. In a larger network with 20 servers, each server might check 19 other servers: there are 380 possible cross-check pairs.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

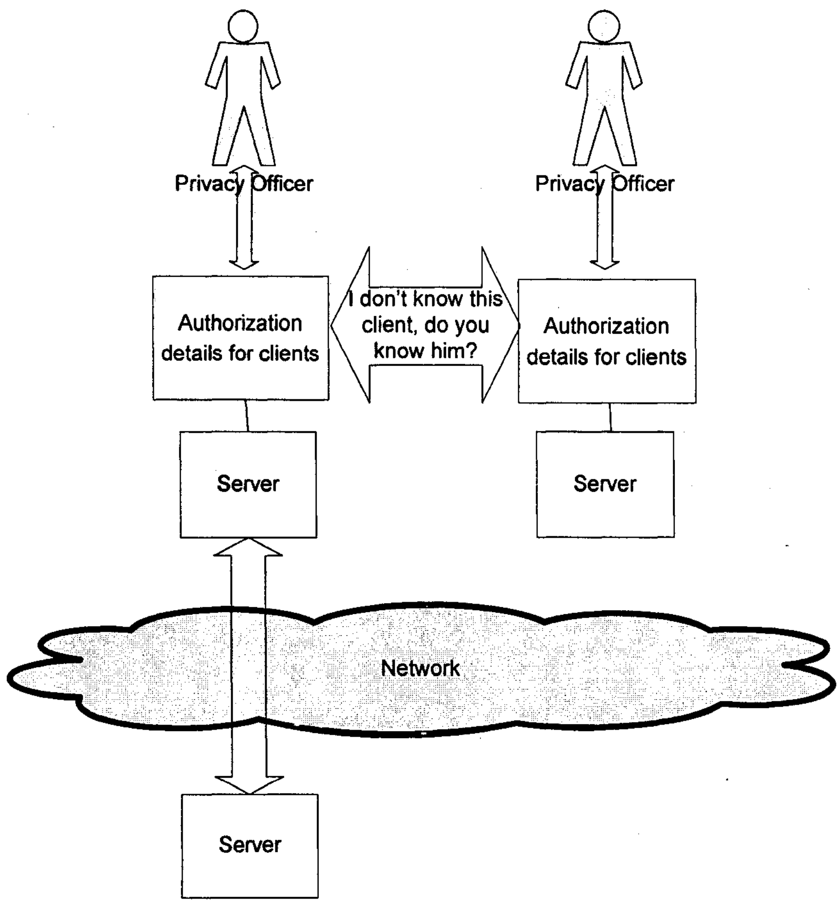

Figure 12 - Inherited Trust

Since this solution has accountability issues and scales poorly, it is unlikely to be a good solution. There may be cases however where it would work for very small networks.

- if there is one privacy control officer (or office) for multiple machines, the accountability issue is much more manageable as there is only one office accountable for the entire system
- if the number of machines is likely to stay small; then the scalability is unlikely to become an issue

The removal of the inherent data duplication problem in the "no collaboration" solution might make this superior to "no collaboration" for very small systems controlled by a single privacy office(r).

The other problem presented by this (and the previous solution) is that it is difficult to

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

approach the concept of purpose without inferring that all data requests to a server are for a single and universal purpose which would be the intent of the service. Adding finer grain purposes (or sub-purposes) will tend to militate against these simple solutions.

3.3.2.1 Accountability Support – Responsiveness

While a Privacy Officer can make a change to a control server to immediately change the behavior of the system, it is much more difficult to make changes to other servers. Because a Privacy Officer can not control the entire system's ability to make decisions, the system is not responsive to any single Privacy Officer.

3.3.2.2 Accountability Support - Absence of Data Duplication

Data Duplication should be fairly rare as the systems can share the data from any one system server. However, there is no easy mechanism to ensure that there is no data duplication; if there is duplicate data for a client or purpose then the system is unlikely to detect it and has no way to resolve the issue.

3.3.2.3 Readability / Simplicity of Privacy Policy

Readability also becomes fairly complicated: a server can publish a statement of its own policies, but would not be able to fully qualify the complete policy because it relies on policies from other servers. Readability would thus be poor.

3.3.2.4 Scalability

In some ways scalability is reasonable; in particular, data duplication can for the most part be avoided. There is no need for duplicate records for clients and/or policy information since, once a server has this information, it is available to the other severs.

There is a major scalability problem with the number of servers that might need to be

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

checked if each server might have to check with every other server. As the number of servers grows, the number of possible checks grows exponentially, resulting in a definite scalability problem.

3.3.2.5 *Network Efficiency*

Network efficiency is definitely a problem for Inherited Trust solutions as any request to other servers can involve requests to several other servers. It is worth noting that the other servers may not have any useful information but may still be activated as part of the request. The extra network traffic and machine activations will definitely result in system inefficiencies.

3.3.2.6 *Bottleneck Avoidance*

While there is not a single resource that could be a bottleneck in the entire system, there might be several. Each server might have other servers waiting on it; each server can hence potentially become a system bottleneck. There is definite potential for the system to become gridlocked.

3.3.3 ***Update Network***

As the service grows it becomes increasingly apparent that the work of the privacy officer needs to be centralized so that the officer can more easily control the system. One approach would be to have a single management tool managed by the privacy officer as shown in Figure 13. This tool could then update the data servers as appropriate when the data changes.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

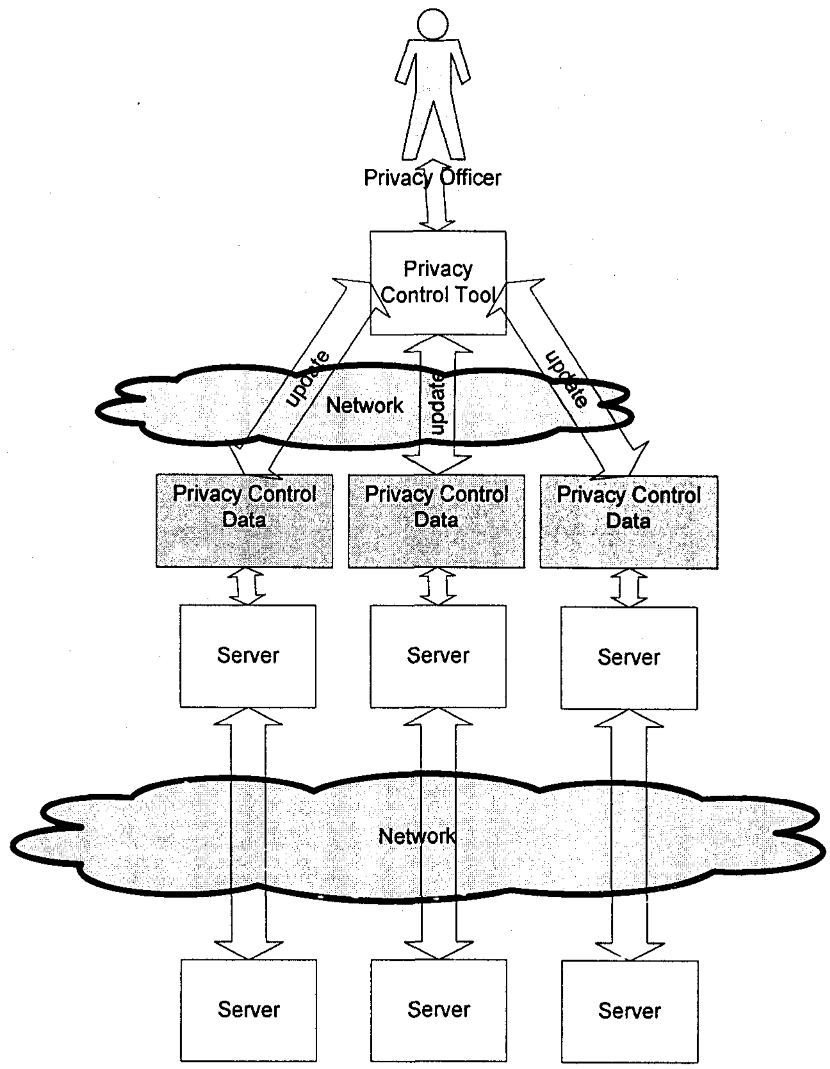

Figure 13 - Update Network

Depending on how frequently updates occur, this would be very similar to the No Collaboration solution but with better accountability and control. However, there are still scalability issues, and they are much direr. The problem stems from the size and frequency of the updates, but there are also potential accountability problems if updates are missed.

3.3.3.1 Accountability Support - Responsiveness

Because the Privacy Officer directly seeds the data tables for all the systems, the system will be fairly responsive to his directives. There is an issue though in terms of how

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

quickly the system can respond. There will always be some delay between a change by the Privacy Officer, and there will always be the potential for a data server to behave inappropriately during that delay.

3.3.3.2 *Accountability Support – Absence of Data Duplication*

There are accountability problems with this system for two reasons:

- if an update is missed on a server, it may make inappropriate decisions. There are several network solutions to make sure that updates will arrive, but there is still the possibility of updates arriving late.
- even with regular updates there will be some non-zero delay between the Privacy Officer making a change and the servers being able to act on it

3.3.3.3 *Readability / Simplicity of Privacy Policy*

Readability should be quite good, as there is a single privacy policy that can be used to create a readable policy statement.

3.3.3.4 *Scalability*

Scalability is complex for this solution, but generally this solution would scale well:

- The individual servers are scalable, as they are almost entirely independent of any other system resource; the data servers act as if there was no collaboration and can handle all requests locally.
- There is a scalability issue for the entire network as a large amount of data will need to be transmitted from the Privacy Officer to all the servers. As the size of these updates increases and as the number of servers needed to be reached increases, the amount of information to be transmitted will grow rapidly and will likely eclipse the amount of useful traffic on the system's network. However, there are several protocols for broadcasting large amounts of information to many end users; so

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

potential good solutions exist.

3.3.3.5 Network Efficiency

While efficiency of any single client request should be quite good, there will be tremendous network resources requirements in broadcasting periodic updates from the Privacy Officer to the servers.

3.3.3.6 Bottleneck Avoidance

There is a central system resource, but as it is not required to handle a client request the system is unlikely to ever become congested.

3.4 Centralized Control Solutions

The previous solutions avoided centralizing the privacy control data. However, those solutions would either suffer accountability problems or efficiency problems. These problems arise because the Privacy Officer does not directly control the Privacy Data: it is either distributed or a copy of the Privacy Officer's data.

The following solutions avoid these problems by requiring the network to access the Privacy Officer's data directly. All the system privacy data is centralized, so that the Privacy Officer can more easily control it.

3.4.1 *Embedded Privacy Control*

The most obvious way to use a centralized solution is to embed a privacy query into the client request. When a client makes a request for private information, the server can verify with a community Privacy Control Component to verify that the client is authorized to receive the information. This method is shown in Figure 14.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

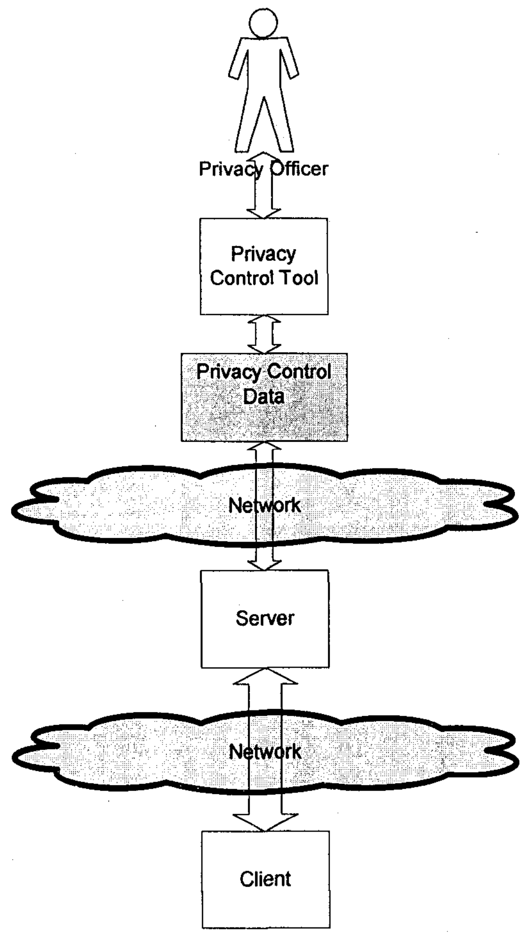

Figure 14 – Embedded Privacy Control

3.4.1.1 Accountability Support – Responsiveness

The responsiveness advantages should be fairly obvious; the Privacy Officer has a local machine that controls all the privacy policies for the community. If the Privacy Officer makes a change in the privacy control data then the change will affect the next client request. The system is extremely responsive.

3.4.1.2 Accountability Support - Absence of Data Duplication

As there is only one data set in the entire community (in the Privacy Control Component), there should be no potential for data duplication. The only potential for Data Duplication would be if the servers inappropriately tried to learn how to predict the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

decisions of the Privacy Control Component, which should be avoided.

3.4.1.3 Readability / Simplicity of Privacy Policy

As the system is controlled by a single Privacy Officer who has a local copy of the privacy control data, it should be fairly straightforward to extract the privacy rules.

3.4.1.4 Scalability

Since there is a single Privacy Control Component (PCC) for the entire community, the system is limited to the amount of privacy verifications that the component can process. The system will not be able to scale past the ability of the PCC to verify requests.

3.4.1.5 Network Efficiency

There is definitely some network inefficiency in this system. Each client request requires that the data server make a further request to the Privacy Control Component. While doubling the number of network messages is not extremely inefficient, more efficient solutions exist.

3.4.1.6 Bottleneck Avoidance

This solution is very prone to gridlock. The Privacy Control Component is invoked for each client request, so the entire system is limited to the number of requests that can be handled by the Privacy Control Component. The Privacy Control Component can definitely become a system bottleneck.

It is reasonable to assume that privacy information requests will be bursty, that a client will make several requests from a server in a short period and then not make further requests for some time. The next method (Session Based Control) will build on this assumption to help alleviate this potential bottleneck.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

3.4.2 *Session Based Control*

As an alternative to Embedded Privacy Control where privacy verification is undertaken as part of each request; Session Based Control provides privacy verification before allowing requests. In this solution, the client first gets an approval ticket from the Privacy Control Component (a ticket generating service) and then he can make requests to a data server. These two steps are shown in Figure 15and Figure 16.

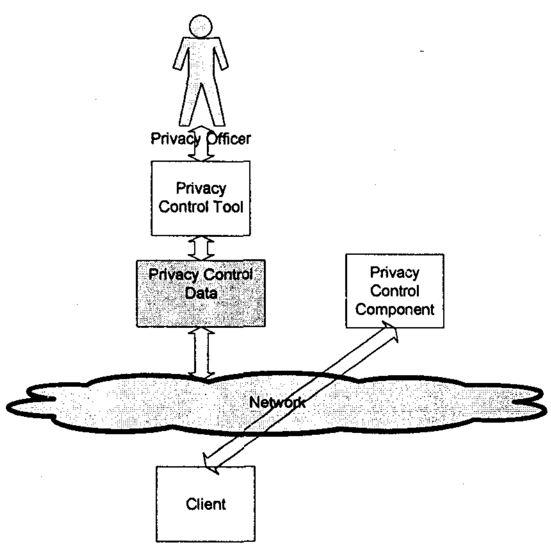

Figure 15 - Session Based Control (Contact PCC)

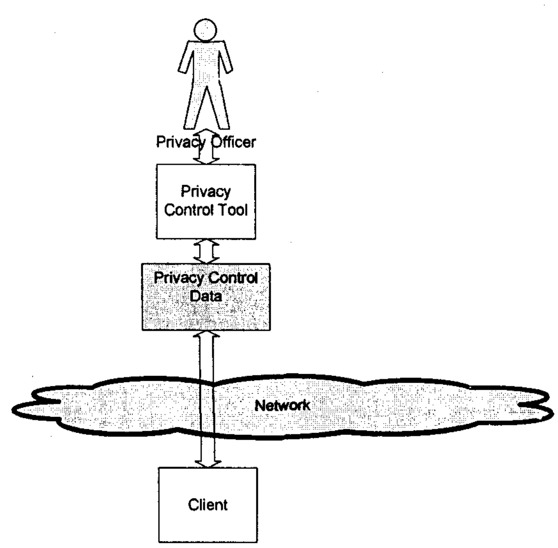

Figure 16 - Session Based Control (Contact Data Server)

One of the potential problems with this solution is that the system would be unresponsive. A client could use one ticket forever even though the Privacy Officer might change the client's permissions. In order, to avoid this problem the approval tickets would have to have a time expiry mechanism. Further, by implementing a time expiry mechanism, a new benefit can be realized: the client can make repeated use of a single ticket. By reusing an approval ticket, the client increases system efficiency by reducing the number of Privacy Control Component invocations.

3.4.2.1 *Accountability Support – Responsiveness*

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Immediately after the Privacy Control Component data is changed the approval tickets being created will follow the new rules. However, there can be tickets in the system. generated with an older data set, that have not yet expired. The Time to Expire concept is required to assure that tickets will eventually become obsolete so that the system has reasonable responsiveness.

The responsiveness of the system can by tuned: lowering the expiry time parameter increases system responsiveness.

3.4.2.2 Accountability Support - Absence of Data Duplication

There is only a single data set in the system, the data set in the Privacy Control Component. There should be no potential for data duplication.

3.4.2.3 Readability / Simplicity of Privacy Policy

There is only a single data set in the system; it should be straightforward to pull the policies controlling system privacy from that data set and create a readable explanation of the system. Later in this thesis, section 6.2.4, a privacy map will be shown and that privacy map contains all the required information to explain how privacy control decisions will be made.

3.4.2.4 Scalability

Because there is a single Privacy Control Component for the entire community, the system is limited to the amount of requests for sessions that the component can process. The system will not be able to scale past the ability of the PCC to create connections.

Note, that this is a significant improvement over the Embedded solution because several client requests can be processed with a single approval ticket from the PCC. If it is

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

reasonable to assume that most clients will make multiple requests from a single server then this system scales reasonably well.

Ticket expiry times will also play a significant part in how scalable the system is. When a ticket expires the client will need to request a new ticket if it wishes to continue making requests of the server. By increasing the expiry time , the volume of replacement ticket requests will be reduced which will allow more new connections to be established.

The responsiveness of the system can by tuned: increasing the expiry time increases system scalability.

3.4.2.5 Network Efficiency

Because several requests can be processed with a single approval ticket, the amount of processing and network traffic required for privacy control will be minimized. This scheme should be quite efficient.

3.4.2.6 Bottleneck Avoidance

Session Based Control will have an obvious potential bottleneck: the Privacy Control Component. Each controlled session between a client and a server will require processing in the Privacy Control Component, so the entire system can never have more connections than the Privacy Control Component can create. However, the bottleneck is on the creation of sessions and not on a single client request; so this system should perform better than the Embedded Privacy Control under the given assumption that client requests are likely to be bursty.

It is worth noting that Time to Expire is again an important tuning parameter. The longer tickets are allowed to remain active the less frequently they will need to be refreshed from the Privacy Control Component.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

3.5 Recommendation

Based on the above discussion and consideration, the following table, Table 1 – Architecture Comparison, compares the proposed solutions:

Table 1 - Architecture Comparison

	Responsiveness	Absence of Data Duplication	Readability	Scalability	Efficiency	Bottleneck Avoidance
No Collaboration	Poor	Poor	Poor	Good	Good	Good
Inherited Trust	Poor	Good	Poor	Poor	Poor	Poor
Update Network	Good	Special	Good	Good	Poor	Good
Embedded Privacy Control	Best	Good	Good	OK	Poor	Poor
Session Based Control	Can be tuned	Good	Good	Can be tuned	Good	Can be tuned

* The Table above is subjective, and is not meant to replace the detailed analysis above. This table is provided as partial justification of the decision to pursue research of session based control.

The No Collaboration solution lacks in the privacy related aspects. It has poor accountability and readability support. This type of system should probably not be implemented, but the expectation is that many real world systems do not actually co-ordinate their privacy control; most real world privacy controlled environments could probably be improved with better design.

The Inherited Trust and similar distributed solutions will generally provide poor solutions; they lack in most privacy related aspects and perform poorly. It is unlikely that this type of solution will ever be fruitful.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The Update Network is a good solution for most purposes, but it does require distributing large amounts of data to all the data servers. Efficiency is likely to be a significant drawback for large applications. It is possible that this solution is worthy of future study however as it does perform quite well in all other regards.

Embedded Privacy Control has one major advantage over the other systems as it is the most responsive solution: there is no real delay between the Privacy Officer changing the Privacy Policy and that change affecting the system. However, there are significant technical problems with this solution: it is inefficient, scales poorly and is prone to system bottlenecks.

Session Based Control seems to be a reasonable improvement to Embedded Privacy Control. It has tunable privacy behavior similar to Embedded Privacy Control (a readable system lacking data duplication with tunable responsiveness), but is likely to outperform it in technical aspects (tunable, scalability and bottleneck avoidance with improved efficiency).

After weighting all options, it was decided to pursue the Session Based Control solution for this thesis.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Chapter 4

Related Work

4.1 Introduction

In this chapter, and now that we are familiar with the requirements of the system, we will look in details at some of the related work, and examine their strengths and weaknesses for our application domain. In particular, three existing technologies that were adapted for the development of the proposed Pluto protocol will be presented:

- **Kerberos**

  Kerberos is an authentication protocol that uses a central trusted server to create ticket based sessions between clients and servers. This solution was the model for the Session Based Control architecture.

- **P3P**

  Platform for Privacy Preferences Project , or more commonly P3P, is an existing protocol and language used to describe privacy polices on web servers. While the P3P protocol has been found to be deficient for general usage; the language for describing privacy policies is completely suitable for our purposes.

- **Role Based Access Control**

  Role Based Access Control (RBAC) is a flexible approach to controlling access to a system by associating permissions

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

with roles and then associating users to those roles. Role Based Access control proved to be a useful way to deal with the concept of reasonable expectation described in Section 2.5 " Consent".

Let us now look into details of each of these protocols.

4.2 Kerberos

Kerberos is a network authentication protocol developed at the Massachusetts Institute of Technology (MIT); it has been included as part of many industrial applications such as Mac OS X, Microsoft Windows (2000 and later), and Eudora among others.

Authentication protocols work by using a shared secret; the client and the server are the only entities that know the password. If an unknown entity contacts the server and knows the secret password, the server can reasonably assume that the entity is in fact the client. The problem with password based authentication is that they can be intercepted unless further measures are taken.

The problem solved by Kerberos is proving the identity of a client to a server on an insecure network where any messages might be intercepted. The normal solution to authentication would be clear text passwords, but it would be impossible for a client to send his password without the possibility that it could be intercepted

Kerberos solves this problem by creating a trusted third party server that can create authentication proofs that are secure even if they are intercepted. The Kerberos solution is to find a shared secret other than a simple text password that can be used to prove who sent a message. In Kerberos, the shared secret is an encryption key, and the genius of Kerberos is that that the secret can also be used to secure the message.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

A possible next step for a password system would be to use public key encryption or hashes of the password, but these solutions still can not make use of the expected bursty nature of information queries. Every server would require specific knowledge of every client; which might cause scalability problems on the network.

Kerberos solves this problem by centralizing the trusted third party. Only the Kerberos Server needs to know every client and server.

4.2.1.1 Encryption

The Kerberos protocol is very dependent on private encryption, and while a full discussion on how those keys work would be out of scope for this thesis, it is worthwhile to provide a brief discussion of their workings.

Private encryption methods rely on a key: an encryption key is typically a long string of random characters that are used in an algorithm to secure data. The data are secured by feeding the message to be sent and the encryption key into an encryption algorithm. The algorithm then uses the key to encrypt the message making it difficult to decrypt without knowing the key. This process is shown in Figure 17.

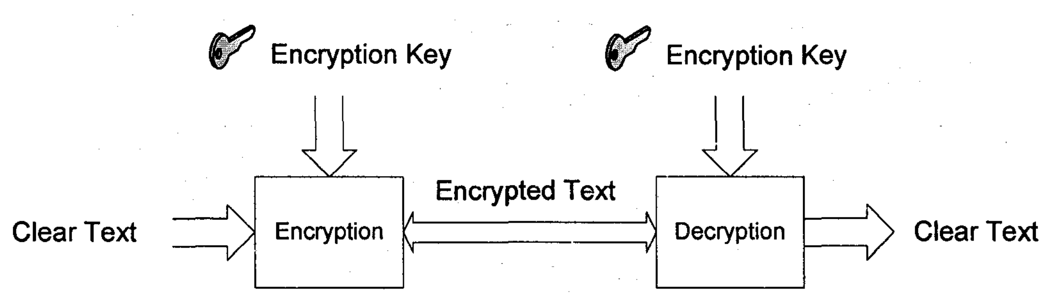

Figure 17 - Basic Encryption

In order, for these encryption routines to work both the encryption and decryption routines will need to have copies of the encryption key. The encryption key is, in fact, a shared secret.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

4.2.2 Kerberos Basics

The Kerberos protocol works by maintaining a special network resource, the Kerberos Server, which holds encryption keys for all clients and servers in the system. These keys can now be used as shared secrets for authentication.

The Kerberos Architecture is shown in Figure 18.

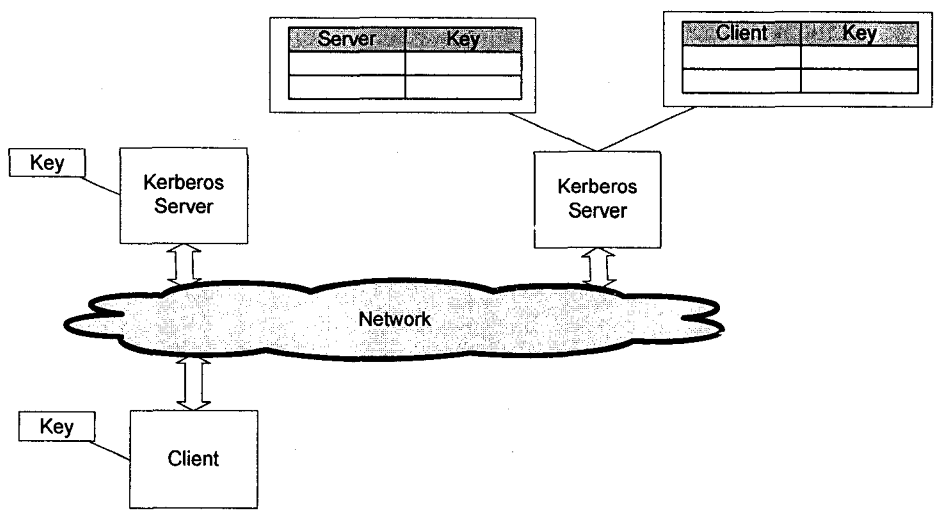

Figure 18 - Kerberos Architecture

When a client wishes to contact a server, it will first contact the Kerberos Server and ask for a session ticket for its preferred server as shown in Figure 19. The server will then create a ticket under the assumption that the client is valid (note: the server has not yet authenticated the client) as shown in Figure 20.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

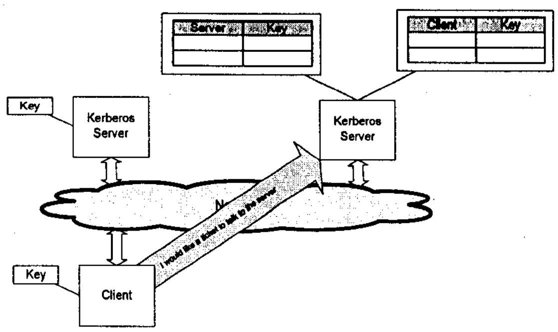

Figure 19 - Kerberos Ticket Request

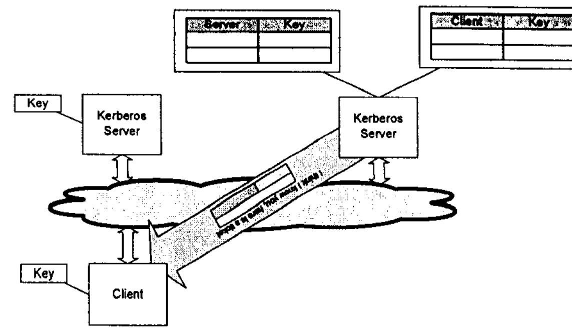

Figure 20 - Kerberos Ticket Response

While there are several important parts of the ticket to be discussed later, the most immediate and important aspect is that the ticket is composed of two major parts: a client portion and a server portion. The client portion and the server portion are both encrypted with the privacy key for that entity as shown in Figure 21.

Kerberos Ticket	
Client Portion (encrypted with Client Key)	Server Portion (encrypted with Server Key)

Figure 21 - Basic Kerberos Ticket

The first thing to note is the client can only decipher his portion if he knows the encryption key that the server used to encrypt the client portion. Because of this, no unauthorized client could pretend to be a legitimate client and make use of the ticket. The ticket is only useful for the legitimate client, and in this way, the Kerberos Server has authenticated that the client is authorized to use the ticket.

Once the client has received the ticket then the client can contact the server and send the ticket to the Kerberos server to open a communication channel as shown in Figure 22. Like the client, the only entities that should be able to make use of the server portion of the ticket are the server and the Kerberos server, so the server can be assured that the ticket was approved by the Kerberos server.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

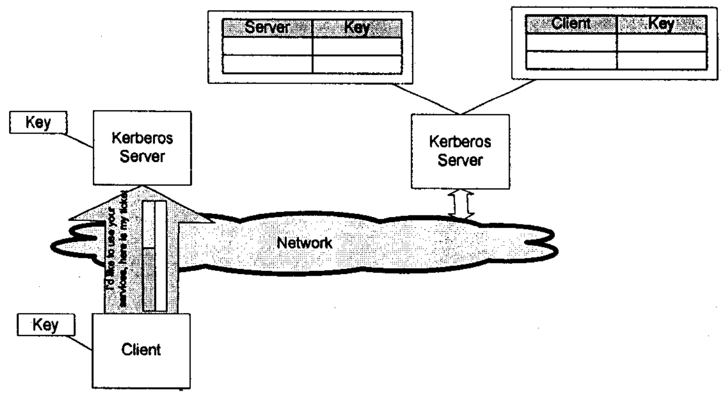

Figure 22 – Kerberos Ticket Sent To Data Server

Both the client and server are thus able to authenticate that the ticket was sent from the Kerberos server, and because the Kerberos server has only generated a ticket for valid clients, the server can also be assured that if the client can make use of the ticket then the client has been authenticated by the Kerberos Server.

4.2.3 Kerberos Ticket Details

There remain some details to ensure that both the client and server are assured that the other entity has been validated. The contents of the client and server portion can be used for that validation.

There are several versions of Kerberos and the largest difference between versions is the contents of the ticket. However, there are several common elements in the Kerberos tickets regardless of versions.

One of the most common and useful data elements in the ticket is a session key: an encryption key that will be allow the client and server to secure their communication. These details have been added to the Kerberos Ticket in Figure 21.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

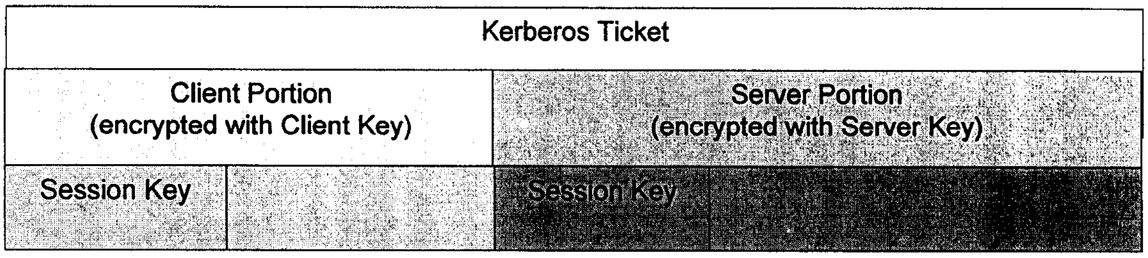

Figure 23 - Kerberos Ticket With Session Key

Since the session key is encrypted within the ticket, it is only accessible to the appropriate client and server; only the client and server will know the session key and in this way the client and server are both assured that any messages using this ticket have been authorized.

Another important part of the ticket is the time to live information. In order to ensure that users are not able to use a server forever, tickets will expire. This is controlled by adding a timestamp and lifespan information to the server portion of the ticket. When a request is made against a ticket, the server will ensure that its local time does not exceed the time to live information in its portion of the ticket. The Time To Live details are shown in Figure 24.

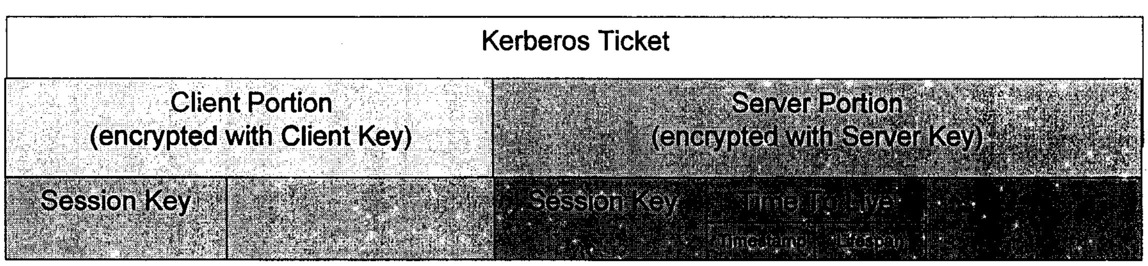

Figure 24 - Kerberos Ticket with Time To Live

Typically the server and client portions will also have authentication details about the client and/or server. Things like the client name and the machine address used to get the ticket can be used for further server validation; similarly the client portion might have authentication details about the server name and/or address. Exact authentication details will vary based on Kerberos implementation. The Authentication details are shown in Figure 25.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

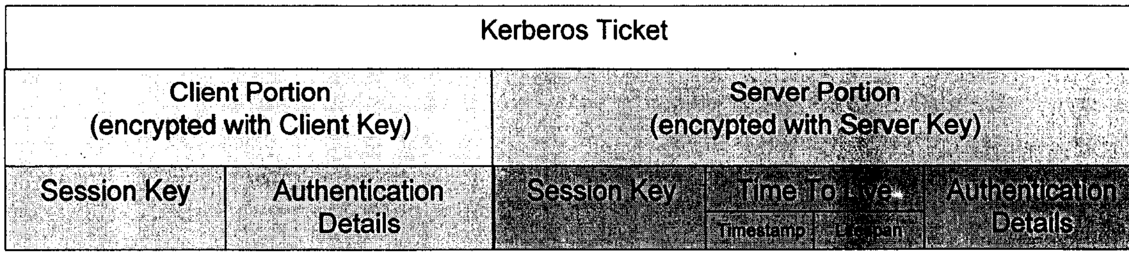

Figure 25 - Kerberos Ticket with Authentication Details

It is possible that additional flags might be included in the Kerberos ticket to indicate various system attributes. However, that level of detail is not required to understand the proposed Pluto Protocol and would likely change if the proposal were to become further established.

4.2.4 Analysis

Kerberos is a authentication protocol. It is designed so that a client can reliably prove its identity to a server (and to prove the server's identity to the client). Kerberos does not however allow for fine grain restrictions on what that client can do on the server.

Privacy requires more detailed control of what is possible; a client might only be allowed (or only want) to have access to some of the data on a server. In order to adapt Kerberos for privacy work, it would be necessary to add context information to Kerberos. That context information can then be used to describe the privacy restrictions on the session. The privacy context details are shown in Figure 26.

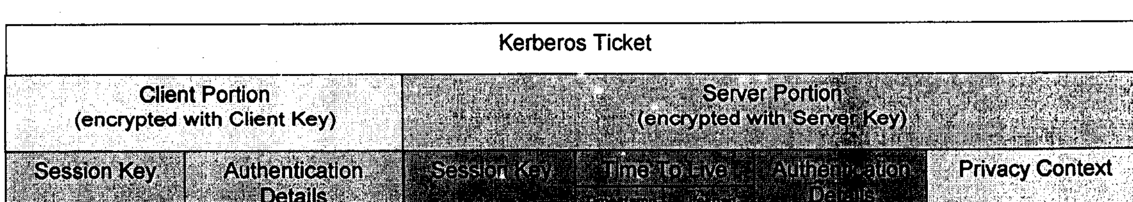

Figure 26 - Kerberos Ticket extended for Pluto

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Note: This thesis does not make specific recommendations for any particular version of Kerberos. Kerberos is a well tested and comprehensive protocol, and as such has occasionally had faults found. This thesis recommends Kerberos's central methodology of using tickets. Further research into specific Keberos versions would be required before moving ahead into industrial deployment.

4.3 P3P

P3P, the Platform for Privacy Preferences Project, is an industry standard for dealing with privacy for web sites developed by W3C, the World Wide Web Consortium. P3P enables web sites to describe how they will use private information to clients who are connecting to use that service. This description can then allow web browsers to automate decisions for personal information exchange.

P3P solves the problem of how does a user's web client (e.g. Firefox or Internet Explorer) decide when it can send personal information to a web site. Web sites will often try to store personnel information in a client's system such as name, passwords, account name etc. so that the web site can reuse this information on return visits or on visits to related websites. Because this information is personal, a client might wish to control or disallow the web site from using that information.

In order to control what information might be released then a client can specify policies on his machine. When he contacts the web site, the client can download the site's policy file to discover how the site will use the client's information. After checking the site's policy against the client's, the client may or may not release personal information to the web server.

4.3.1 P3P Basics

P3P works by adding additional information to web page queries. When a client wishes to retrieve a web page that hosts some sort of service, that web page may specify a

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

potential P3P policy file. If the web page does specify a policy file, the client's web browser can then download the policy file and compare the web page's policy file to the browser's policy file.

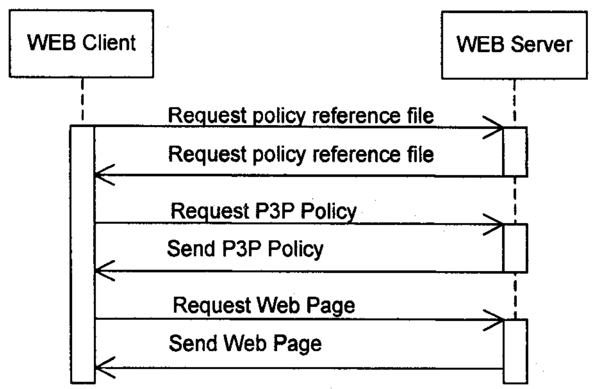

Figure 27 - P3P Sequence

As illustrated in Figure 27, the P3P sequence is:

- The client will ask the web server for a policy reference, and the web server will return the policy file.
- The client can check the policy reference for the location of the P3P policy file and then request the P3P policy file from the web server.
- After getting the P3P policy file from the server the client can now proceeded to download the web page passing appropriate private information.

4.3.2 P3P Meta Language

The success of the P3P protocol is heavily based on the language used by the server to describe the privacy policy. P3P defines a very rich XML based meta language for servers to describe privacy policies. The richness of the P3P policy description language makes it impractical for a complete description in this research; however, a partial understanding will be required to describe the Pluto protocol.

The P3P meta language describes the entity that will be using the private data, and then

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

describes how the data will be used and the purposes for which it is being requested. As an example, the server's policy (in English) might be:

> We do not currently collect any information from visitors to this web site except the information contained in standard web server logs (your IP address, referrer, information about your web browser, information about HTTP requests, etc.). The information in these logs will be used only by us and the server administrators for web site and system administration, and for improving this site. It will not be disclosed unless required by law. We may retain these log files indefinitely. Please direct questions about this privacy policy to *privacy@p3pbook.com*.[1]

This could be expressed in a compact P3P policy as shown in the following P3P file.

[1] Example from page 5, [30]

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
<POLICIES xmlns = "http://www.w3.or/2002/01/P3Pv1">
<POLICY discuri = "http://p3pbook.com/privacy.hml" name
= "policy">
 <ENTITY>
 <DATA-GROUP>
 <DATA ref="#business.contact-info.online.email">
       privacy@p3pbook.com
 </DATA>
 <DATA ref="#business.contact-info.online.uri">
       http://p3pbook.com/
 </DATA>
 <DATA ref="#business.name">
       Web Privacy With P3P
 </DATA>
 </DATA-GROUP>
 </ENTITY>
 <ACCESS><nonindent></ACCESS>
 <STATEMENT>
       <CONSEQUENCE>
             We keep standard web logs.
       </CONSEQUENCE>
       <PURPOSE>
             <admin/><current/><develop/>
       </PURPOSE>
       <RECIPIENT><ours/><RECIPIENT>
       <RETENTION><indefinitely/><RETENTION>
       <DATA-GROUP>
             <DATA ref="#dynamic.clickstream">
             <DATA ref="#dynamic.http">
       </DATAGROUP>
 </STATEMENT>
</POLICY>
</POLICIES>[2]
```

[2] Example continued on page 6, [30]

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

A closer examination of that P3P compact policy statement is useful to understand the parts of the statement:

- Lines 1-3 and 32,33 wrap the statement and identify it as a policy object
- Lines 4-16 identify and describe the web server that hosts and can reasonably use this policy statement.
- Lines 17-31 are the actual policy data. This portion contains a description of the data being requested and the purposes for which it will be used.

Lines 17-31 are the specific privacy policy information that is important to the client's server, and the rest of the file being a template for the site in question. It is worth looking more closely at the policy information section (17-31):

- Line 17 briefly describes the access level.
- Line 18, 31 encapsulates the policy.
- Line 19-21 provides a brief textual description of the intent of the policy.
- Line 22-24 provides a list of purposes
- Line 25 describes who will be collecting, using and retaining the data.
- Line 26 describes how long the data might be kept.
- Line 27-30 describes the exact data that are being requested.

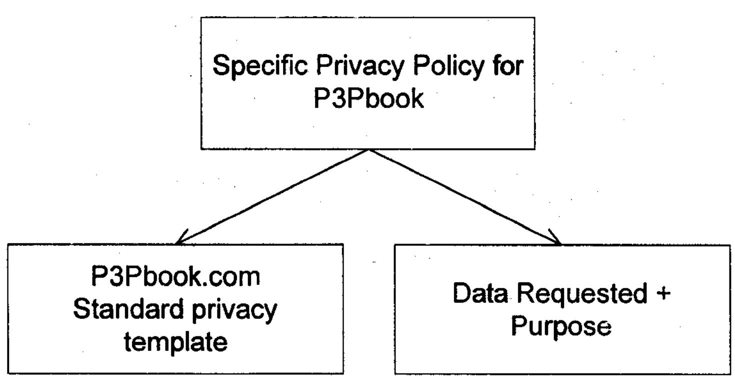

Figure 28 - Decomposed P3P File for pepbook.com

For the most part this is again boiler plate except for a very small amount of information. In particular lines 22-24 provide a list of purposes, and lines 27-30 provide a description of the data that are being requested. It is possible to view this entire document as a

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

standard template for privacy on p3pbook.com, except for the list of purposes and data description. This is shown in Figure 28.

4.3.3 P3P Requirements

While P3P is a complete solution for its problem, that problem requires several assumptions that might not hold in the general case.

Most notably, P3P uses a push model for personal information exchange. The entity that has the information is the entity initiating the exchange. The client is contacting the server, and it is the client that is deciding whether or not to send personal information. Because it is the client who has the information, P3P can make the reasonable assumption that the server is what it claims to be.

In a scenario where the client is asking the server for personal information, P3P can no longer reasonably make the assumption that the client is who he claims to be. P3P does not have any authorization mechanisms to ensure the client's authenticity. P3P can not work as a general solution because there is no way to verify a client's identity.

Further, P3P does not have inherent security. It is assumed that P3P clients will use a secure connection (typically assuming HTTPS) but P3P does not make specific recommendations or requirements. P3P is not a complete privacy solution because it lacks security on its own.

4.3.4 Analysis

While the P3P protocol might not work for general privacy concerned networks, P3P does have a very well developed meta-language for describing privacy policies. The meta-language has been developed both to describe how an entity might use personal information and why an entity might share personal information. The ability to properly describe policies for requesting and sharing information is absolutely required to meet

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

privacy concerns (in particular 2.4 Purpose Based Storage and 2.7 Usage Limitation).

The P3P policy meta-language provides an industry established way to describe privacy policies, and that meta-language can certainly play an important role in any new privacy protocol.

However, P3P policies are not generally created dynamically. The typical case will be to write a policy for a domain once and refer to that single static policy file for all requests for data on that domain. This is unlikely to work in a dynamic electronic community where data requests are likely to be unpredictable. In order, to function in a dynamic environment a good solution for electronic community privacy control should have a mechanism to dynamically create P3P policy files.

One of the other possible criticisms of P3P is that it is too generic; there are too many possible options to capture every possible important policy statement. However, this should not be a major problem in a community environment where specific limitations on the number of privacy statements can be introduced for that community.

The first part of that solution has already been presented. The meta-language description in this chapter has shown that a P3P policy can be regarded as a standard template with room to add specific purpose and data request information. It remains to be shown how that purpose and data request information can be dynamically determined.

4.4 Role Based Access Control

Role Based Access Control (RBAC) is a common access control method for secure networks where system users might have limited access to the system. RBAC has been used in a variety of projects including Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA) security specification and projects by the U.S. National Security Agency (NSA) and U.S Internal Revenue Service (IRS).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

RBAC solves the problem of how to create a user intuitive way to map permissions to users. RBAC relies on the idea that users can be grouped into types (roles) by their job in the system: administrator, user, client, auditor, etc.

Instead of adding permissions to a user directly, RBAC administrators can only add permissions to roles. Users in the system can then be assigned roles, and acquire the permissions associated with that role.

4.4.1 RBAC Basics

Role Based Access Control uses three primary abstractions: users, roles, and permissions. Roles are granted permissions and users are associated to the role based on the job they have in the system.

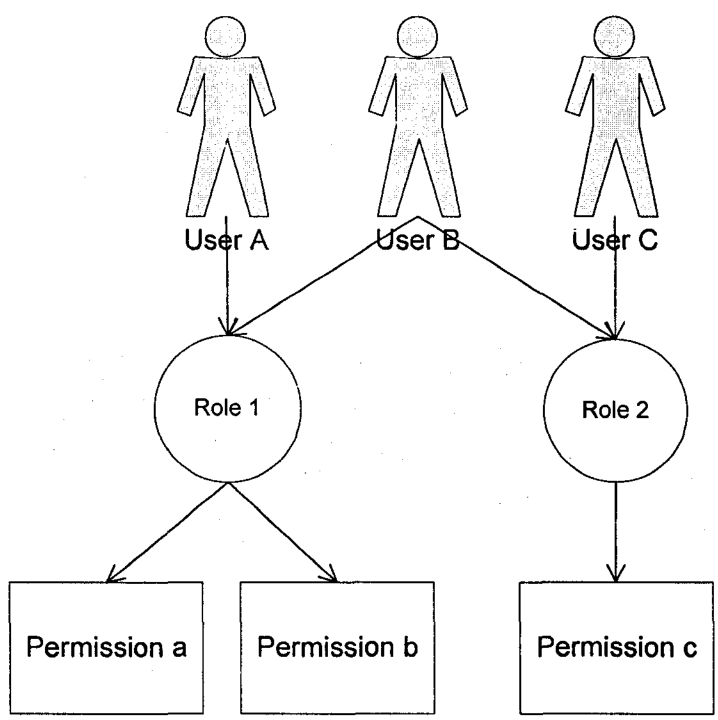

Figure 29 - Role Based Access Control Example

In order to find the permissions a user has, the system can trace from the user to all roles and permissions descended from that user in the privacy map and then aggregate the permissions for those roles. This aggregated set of permissions is the valid permission for

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

that user.

In Figure 29, User B is associated to Role 1 and Role 2, the aggregate set of permissions for those roles is Permission a, Permission b, Permission c. So User B can validly use the system for Permission a, Permission b, and Permission c.

4.4.1.1 Role Hierarchies

One enhancement common to RBAC systems is the use of role hierarchies to assist in the capturing of moderately complicated models. This addition allows roles to inherit all the aggregate permissions of another role.

For example, a super user will generally have all the permissions of a normal user plus some additional permissions. Instead of linking all the normai user permissions to the super user role, the super user is instead linked to the normal user role and inherits all those roles.

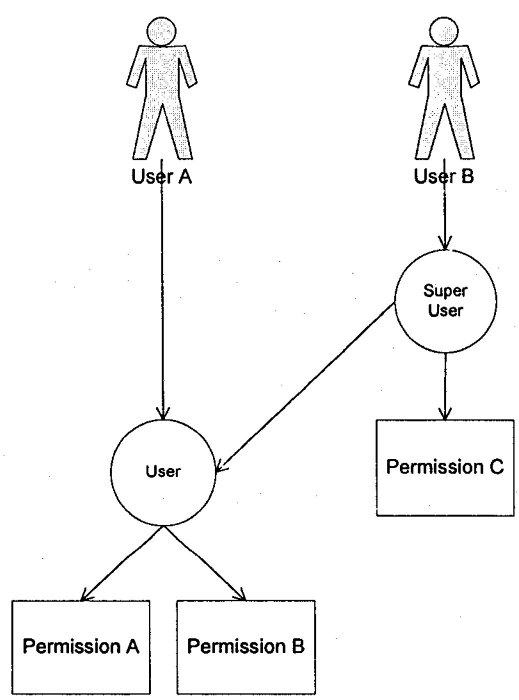

Figure 30 - Role Based Access Control, Role Heirarchy Example

Use of hierarchies has several benefits. Most notably the use of hierarchies streamlines

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

the model and makes maintenance easier. The model is streamlined because there will typically be far fewer connections in the model; if the user role has 3 permissions then the super user account has only one connection to the user role instead of 3 connections to the permissions. Perhaps more useful is the assistance this provides to maintenance; if a new permission is added that should be available to both users and super users only the user role would need to be updated.

4.4.2 Analysis

In the P3P discussion there was mention that there was need to dynamically add purpose information to a template P3P XML file for purpose and data information. RBAC can be used as the primary logic engine for assigning those purposes and data type information. It is probably worth looking again at Figure 28 presented in 4.3.3 P3P Meta Language (and shown again below as Figure 31):

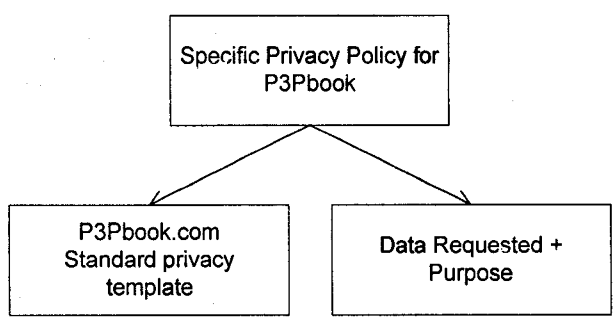

Figure 31 - Decomposed P3P File for pepbook.com (reprise)

This P3P privacy policy file can be seen as a merger of two files, the standard privacy template which is associated with a user and the Data Requested and Purpose information that needs to be dynamically built for a specific private information request.

The P3P message can be built from a slightly modified RBAC model by adding the template information to each user account, and encapsulating the data request + data purpose information into the RBAC purpose.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

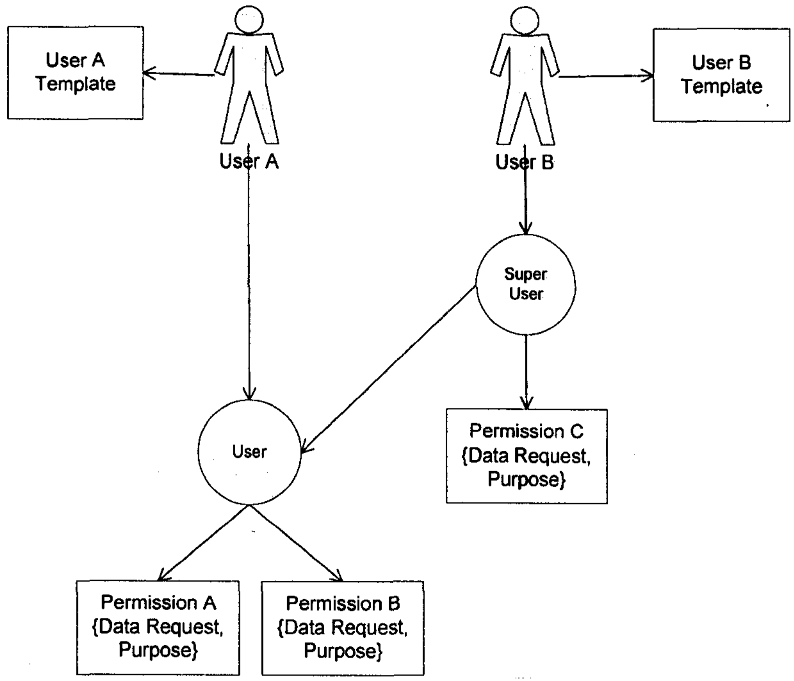

Figure 32 - Role Based Access Control for P3P Example

The exact rules for building the P3P xml file from these sub-files will follow in the next chapter when the Pluto protocol is revealed.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Chapter 5

Pluto: The Proposed Protocol

5.1 Introduction

Having explained the requirements and supporting technologies that will be incorporated into the proposed protocol, it is now possible to reveal the proposed Pluto Protocol. The Pluto Protocol has been designed to meet all of the legal requirements for privacy control, as well as meeting the design goals discussed in Chapter 3.

> The Kerberos protocol is named for the mythological three headed hound that guards the Greek Underworld. Kerberos guarded the living from the dead, a form of access control for the afterlife.
>
> In that mythology, Pluto (also known as Hades) was the god of the Underworld where the dead were judged and assigned their ultimate fate. Pluto, the protocol, is much like Kerberos as it can limit user access; but Pluto can also judge a user and limit its access.

The Pluto Protocol is based in large part on the Kerberos protocol. but adding in the concept of context. This context information is encoded in the P3P Meta Language that is dynamically created using Role Based Access Control.

5.2 Protocol Details

The ultimate goal of the Pluto protocol is to create a privacy controlled session that will allow authenticated clients to access information on data servers while still maintaining appropriate privacy control.

The power to create privacy controlled sessions is held by a network resource, the Privacy Control Component which is directly controlled by the Privacy Officer for the community. The general architecture of the Pluto system is shown in Figure 33.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

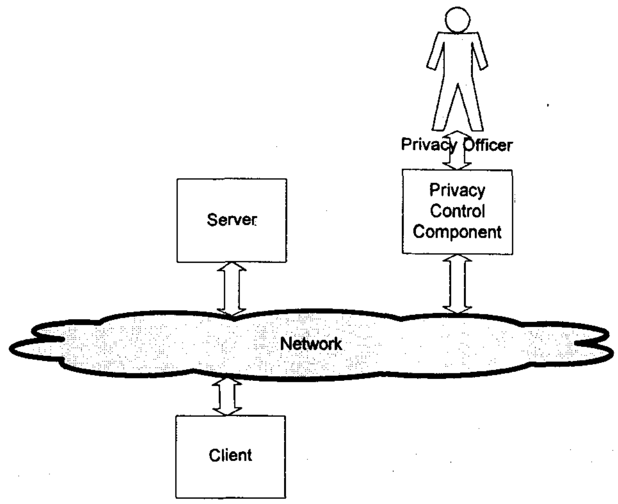

Figure 33 - P3P Architecture

The typical request flow, as shown in Figure 34, will consist of:

- the client contacts the Privacy Control Component (PCC) and requests a session ticket
- the Privacy Control Component ensures that the request is valid, and if it is then the PCC will create a session ticket and return it to the client.
- the client will ensure the ticket is appropriate
- the client will then pass the ticket to the server to establish a session
- the data server will accept, verify and store the ticket
- the client can now send a request to the server
- when the server receives a request it will retrieve its copy of the session ticket and extract purpose information from it. The server can then create a response to the request that is filtered according to the session permissions.

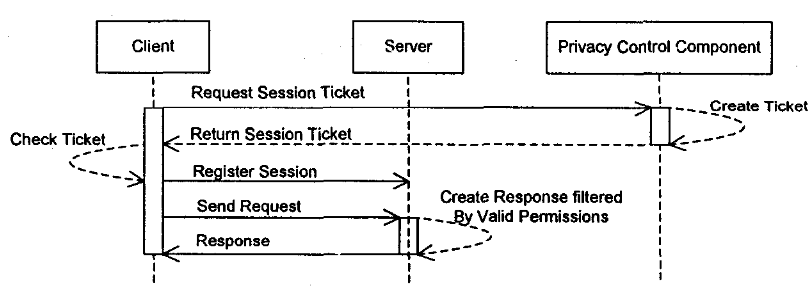

Figure 34 - Pluto Sequence

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

5.2.1 Setup Requirements

Before a client is able to create a Pluto privacy controlled session with a server, the system requires some setup. These setup requirements are managed by the Privacy Officer and require an external method for the client and servers to communicate purpose information and encryption keys with the server.

The methods for exchanging keys and negotiating purpose restrictions will not be covered here as it is likely that this exchange will differ greatly across implementations. While the detail of how this information is exchanged is outside the scope of the protocol a description of the information required for the protocol to work is required.

5.2.1.1 Client Setup

In order to communicate with the PCC and use tickets at all, the Client will need to have exchanged an encryption key with the PCC. Further, because the client will ask for a ticket for a specific role, the client will need a known list of roles for which the client might be given a ticket.

The client will also need to know the names (and likely the address) of the server that the client wishes to connect with. There are a variety of ways this server name might be determined; it may be hard-coded, input directly by the user, found using a lookup server, or retrieved from the server using a special name function on the server. Regardless, the client will need to determine the name of the server to be connected because the PCC will be creating a session for a specific server.

5.2.1.2 Server Setup

Clients will be sending Pluto tickets to servers (the tickets will be generated by the PCC and then sent to the client who will pass them to the server), and those tickets will have information that is encrypted with the server's encryption key. So the server will need to

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

know its key for the privacy community.

Further, servers may not want to release all of its private information to the community, thus the server can restrict the permissions that it grants a community. If the server is restricting community permissions, then it will want to maintain a list of valid purposes for the community in order to verify incoming Pluto tickets.

5.2.1.3 Privacy Control Component Setup

The PCC will need to know each client and server and reflect the setup information in those entities. For each client, the PCC will need to maintain an encryption key and a list of possible roles for which that client can validly request tickets. For each server, the PCC will need to maintain an encryption key and a list of permissions that that server has granted to the community. The PCC will also need to store a P3P template for each server into which will be inserted P3P statements that can define the P3P policy for the session.

The PCC will also need to organize a privacy map that organizes roles and permissions. The roles available to each client and the permissions granted to the community by each server will be associated to the permissions in this map. An example of this privacy map is shown in Figure 35.

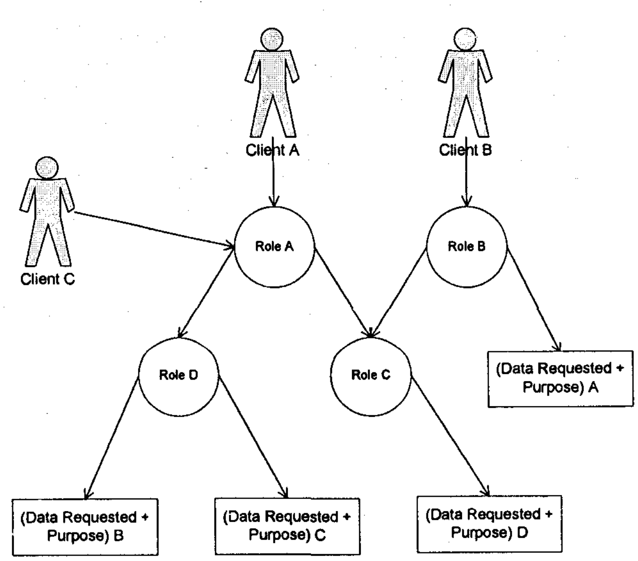

Figure 35 - Example Privacy Map

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

5.2.2 Detailed Sequence Step

Given the above overview, it is now possible to more closely examine each step. Provided below is a brief description and pseudo-code for each step.

```
Ticket RequestTicket()
{
      Client client = getClient(clientName);
      Role role = getRole(requestedRole);
      Server server = getServer(serverName);
      RequestDetails details = RequestDetails (role, server);
      Byte[] encryptedPortion =
            Encryption.Encrypt(details.ToString(),
            this.pccEncryptionKey);
      try{
            Ticket retrievedTicket =
                  RemotePrivacyControlComponent.RequestTicket(
                        client.ToString(),
                        encryptedPortion.ToString());
            ...
```

5.2.2.1 Client Requests A Ticket

In order for a client to create a secure session, the client will need to request a ticket from the PCC. In order to do this, the client will create a ticket request, the ticket request will have the client's name, the client's intended role, and the name of the server the client wishes to connect with. The role and server name should be encrypted with the client's password; the client's name is transmitted as clear text (possibly be encrypted using public key encryption in future versions).

```
Ticket RequestTicket (string clientName, byte[] requestDetails)
{
      Client client = Clients.getClientByName(clientName);
      String clientsEncryptionKey = clientKeys(clientName);
      RequestDetails details = Encryption.Decrypt(
            requestDetails, clientsEncryptionKey);
      Role requestedRole = details.getRole();
      Server server = Servers.GetServerByName(
            details.getServerName());
```


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

5.2.2.2 Privacy Control Component Creates a Ticket

When the PCC gets a request for a ticket from a client, it will first look up the client by name in the PCC's list of clients. With the client record retrieved it can decrypt the role and server name.

With the client's desired role and the name of the server the PCC can determine the permissions (and type of data) that should be used to define the limits of the session. In order, for a permission to be granted:

- the permission (and type of data) must be a child of the role the client specified when requesting the ticket
- the permission (and type of data) must be a child of one of the roles the client has been granted in the community
- the permission (and type of data) must have been granted to the community by the data server

This is shown graphically in Figure 36.

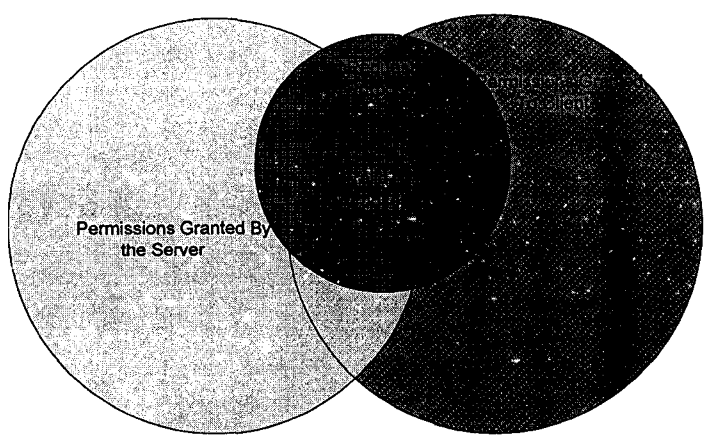

Figure 36 - Valid Purpose Venn Diagram

In order to determine the set of permissions that are descended from the requested roles, and client roles, the PCC will traverse its Privacy Map (a directed graph consisting of roles and permissions) starting at specified roles associated with the client or the requested role. The set of permissions allowed by each server is stored within each server. The intersection of those sets is the set of valid purposes; if that set is empty then

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

there are no valid permissions and no ticket should be generated.

```
...
PermissionSet requestedPermissions =
      PrivacyMap.GetPermissionsDescendedFromRole(requestedRole)
;
PermissionSet clientAllowedPermissions =
      PrivacyMap.GetPermissionsDescenedFromRoles(
            client.allowedRoles);
PermissionSet serverAllowedPermissions =
      Server.allowedPermissions;

// valid permissions are the intersection of requested,
// client allowed, and server allowed permissions
PermissionSet validPermissions = PermissionSet.GetIntersection(
      requstedPermissions,
      clientAllowedPermissions,
      serverAllowedPermissions);
if (validPermissions.Count == 0)
{
      // return no ticket
      ...
}
```

After the intersection of the requested client allowed and server allowed permissions has been found, the valid purposes are defined and a specific permission generated. This permission will include information about what data can be sent with that permission. The final list of permissions and data types can be used to create a P3P statement description to describe the data and permissions that are valid. This illustrated in Figure 37.

```
...
P3PTemplate template = server.p3pTemplate;
template.InsertPolicyStatements(validPermissions);
String p3pText = template.GetP3PStringDescription();
...
```


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

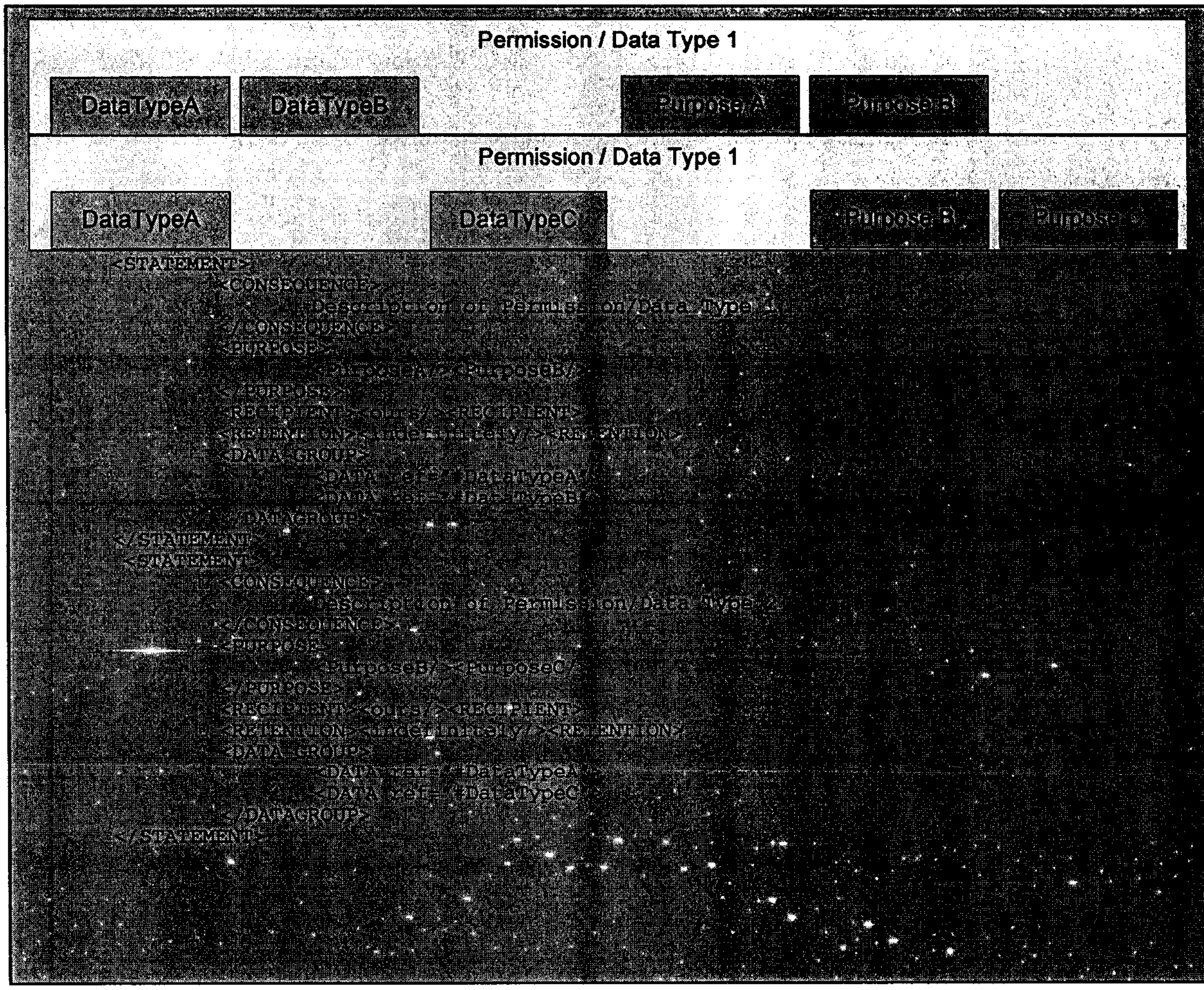

Figure 37 – Example P3P Statement Creation

Because there may be several permissions with different purposes and data types specified, the P3P statement is likely to have multiple sub-statements but that is not an issue.

After determining the statements, the P3P template for each server can be determined from the server's record. The statements can be inserted into the Template to create a complete P3P file to define the policy statement for the session.

Once the policy statement has been defined, ticket creation will proceed as if the ticket were a standard Kerberos ticket except that the P3P policy statement will be added to the server part of the ticket. Other than the addition of the P3P policy statement, a Pluto ticket is a Kerberos ticket including timeout information, and identifier information.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
    ...
    PlutoTicket createdTicket = PlutoTicketFactory.CreateNewTicket
          (client, server, p3pText);
    return createdTicket;
}
```

5.2.2.3 Client Checks Ticket

After a ticket has been generated by the PCC and returned to the client, the client will simply test to make sure that a ticket has been generated. If no ticket was created, then the client can generally assume that there are no valid purposes for which it can request data from the server.

```
Ticket RequestTicket()
{
      Client client = getClient(clientName);
      Role role = getRole(requestedRole);
      Server server = getServer(serverName);
      RequestDetails details = RequestDetails (role, server);
      Byte[] encryptedPortion =
            Encryption.Encrypt(
                  details.ToString(),
                  this.pccEncryptionKey);
      Ticket retrievedTicket = null;
      try{
            retrievedTicket =
                  RemotePrivacyControlComponent.RequestTicket(
                        client.ToString(),
                        encryptedPortion.ToString());
            if (retrievedTicket == null)
            {
                  // assume no valid permissions
            }
            return retrievedTicket
      }
      catch(Exception e) {// handle other exception}
```


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

5.2.2.4 Client Passes Ticket to Data Server

Having retrieved a ticket from the PCC, the client can now pass the ticket to the Data Server to initiate the session.

```
void InitiateSession(RemoteDataServer dataServer, Ticket
pccTicket)
{
	try{
		RemoteDataServer.CreateSession(pccTicket);
	} catch()
	{
		// the server rejected the ticket
	}
}
```

5.2.2.5 Data Server Accepts Ticket

When the data server is given a ticket because a client would like to create a session with it, the first thing the server will do is verify the ticket. Primarily the server wants to ensure that the permissions included in the ticket are permissions that the client has given to the community, but there are other standard Kerberos verification requirements such as ensuring that the ticket has not yet timed out, etc.

If the ticket is valid, the ticket will be added to a collection of tickets for open sessions.

```
void CreateSession(Ticket pccTicket)
{
	try{
		PlutoTickets.Verify(pccTicket);
		OpenSessions.Add(pccTicket.ClientName, pccTicket);
	} catch()
	{
		// handle verification error
	}
}
```

5.2.2.6 Client Sends a Request

After the client has created a session it can create a request. The body of the request should be encrypted with the session password that is included in the ticket (and more specifically in the client's portion of the ticket). The message can then be sent to server.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
string DataServerRequest(String requestDescription,
      Ticket sessionTicket, Server server )
{
      byte[] encryptedRequest =
            Encryption.Encrypt(
                  requestDescription,
                  sessionTicket.sessionKey);
      try{
            byte[] response = server.MakeServerRequest(
                  this.clientName, encryptedRequet);
      ...
```

5.2.2.7 Data Server Responds to Request

When the Data Server receives a request, it will need to retrieve the ticket from the open sessions. Once the ticket has been retrieved, it should be verified again as data (in particular the time) will have changed since the last time the ticket was used. If the ticket is still valid, the request can be handled.

The request will be handled and all data will be returned using the permission set included in the ticket to ensure that no inappropriate information is sent. After building an appropriate response, the server will encrypt the response and return it to the client.

```
byte[] MakeServerRequest (String clientName,
      byte[] encryptedRequest)
{
      Ticket sessionTicket = null;
      try{
            sessionTicket = OpenSessions(clientName);
            PlutoTickets.Verify(sessionTicket);
      } catch()
      {
            // the ticket is invalid
            // send a rejection to the client
      }
      string requestText = Encryption.Decrypt(
            encryptedRequest, sessionTicket.sessionKey);
      string response =this.HandlePrivacyFilteredRequest(
            requestText, sessionTicket.permissions);
      byte[] encryptedResponse =
            Encryption.Encrypt(
                  response, sessionTicket.sessionKey);
      return encryptedResponse;
}
```


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

5.2.2.8 Client Receives Response

Assuming a proper response was returned (and not an exception) then the client can decrypt the response and continue.

```
string DataServerRequest(String requestDescription,
      Ticket sessionTicket, Server server )
{
      string response = null;
      byte[] encryptedRequest =
            Encryption.Encrypt(
                  requestDescription,
                  sessionTicket.sessionKey);
      try{
            byte[] encryptedResponse = server.MakeServerRequest(
                  this.clientName, encryptedRequet);
            response = Encryption.Decrypt(
                  encryptedResponse,
                  sessionTicket.sessionKey);
      } catch
      { // handle exception }
      return response;
}
```

5.3 Pluto Protocol vs. Privacy Requirements

Having presented the Pluto Protocol in the previous section, it is now reasonable to verify that it meets the privacy criteria that were presented in Chapter 2.

5.3.1 Accountability

The Pluto protocol has been designed to centralize decisions for release of data in the PCC that is directly controlled by a Privacy Officer. The majority of decisions are made by the PCC and are based on properly setting up the Privacy Map to automate those decisions.

Looking back at the technical suggestion made in section 2.3 – Accountability, it is possible to recognize the Privacy Controlling Rules as the PCC. Pluto follows the model given in Figure 1 (repeated below as Figure 38).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

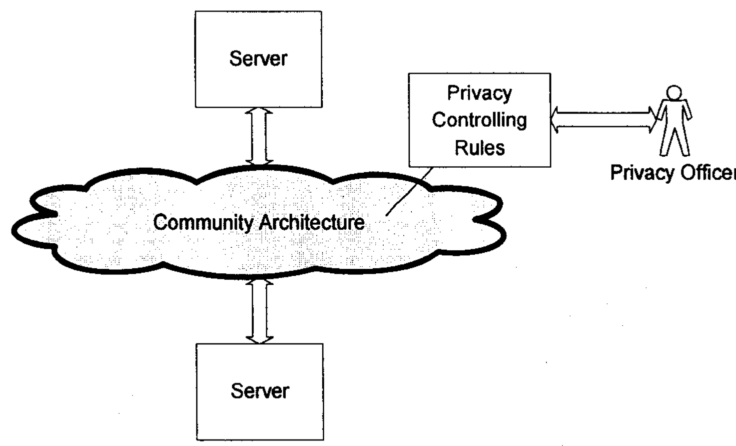

Figure 38 – Requriments for Accountability (reprise)

5.3.2 Purpose Based Storage

While it is somewhat outside the scope of a community protocol to formalize how the data servers will store their data, Pluto does assume that the data servers can make reasonable data release decisions based on the contents of the P3P statement in the Pluto session ticket.

Regardless of what decisions the data servers make in their storage of data, the Pluto protocol requires that all data servers understand the concept of reason and only release data for appropriate purposes.

5.3.3 Consent

Again, it is outside the scope of a general community protocol to formalize restrictions on data servers. The community will have to assume that data on the data server have been collected appropriately.

It is worth noting however that Pluto does handle the "reasonable expectation" concept quite well. The issue was that some purposes could be seen to be a "reasonable expectation" of another parent purpose. Pluto can be seen as fulfilling the requirements illustrated in Figure 3 (repeated below as Figure 39).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

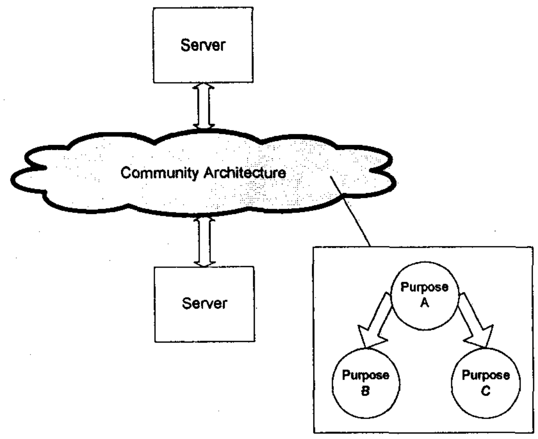

Figure 39 - Requirements for Consent (reprise)

Pluto handles this by introducing the RBAC based Privacy Map; while the RBAC system does not directly associate purposes it does something very similar by introducing roles. One of the things that was noted during early discussion about privacy was that purposes don't necessarily have "reasonable expectations;" when trying to describe how a sub-purpose was a "reasonable expectation" of another purpose the description was always about jobs, duties or roles.

Instead of creating a simple purpose map, roles were introduced as a tool to organize and better describe how purpose might be "reasonable expectations" of other purposes. This privacy map was shown in Figure 35 (repeated below as Figure 40).

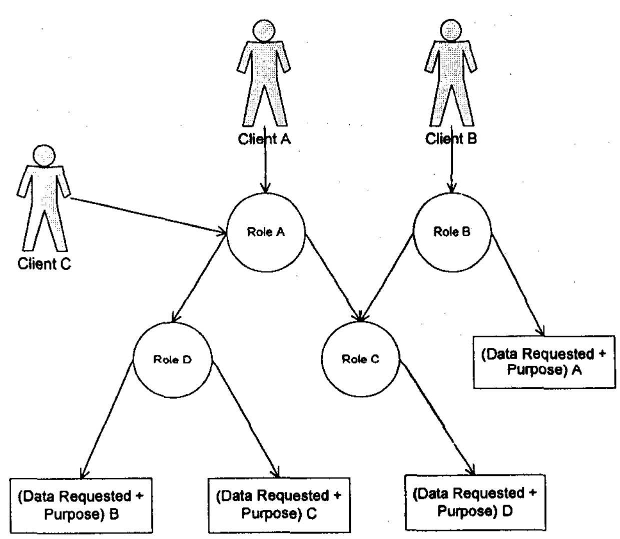

Figure 40 – Example Privacy Map (reprise)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

5.3.4 Collection Limitation

How data are originally entered on a data server is outside the scope of community control; Pluto does protect any further data collection by clients who might be getting the data from a data server. In order to gain any information the client will need to create a session, and the session will have a P3P described purpose statement that defines what information is available from data servers. Pluto does meet the requirements shown in Figure 6 (repeated below as Figure 41).

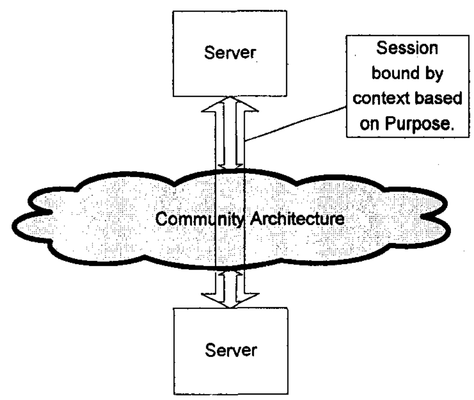

Figure 41 - Requirments for Collection Limitation (reprise)

5.3.5 Usage Limitiation

While the protocol to retrieve data can't entirely keep a client from using data inappropriately, Pluto does require that the client commit to a purpose when the client requests information. A client could request data for one purpose and then use it for another, but, the client cannot easily claim that it was done without knowing what was appropriate. Pluto does fulfill the requirements illustrated in Figure 7 (repeated below as Figure 42).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

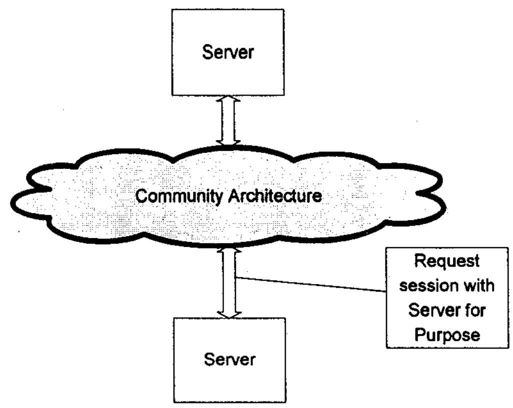

Figure 42 – Requirements for Usage Limitation (reprise)

5.3.6 Accuracy

Accuracy would be difficult to directly enforce. However, Pluto shows promise in helping. In particular, Pluto automates the decision process required to release private information in a community. By streamlining and automating the process to get data from servers, Pluto will enable clients to acquire up to date data from the server.

5.3.7 Safeguards

Pluto is based on a known and very secure protocol, Kerberos. The only information that is ever clear text crossing any network is client names, and those are only useful if the client knows the secret encryption key(s) associated with that name. Private data is very well protected in the Pluto protocol.

5.3.8 Open Policy

The discussion in Section 2.10 – Open Policy identified two forces that would enable a privacy controlled system: simplicity and understandability. The Pluto protocol was designed to meet these requirements. All of the complexity in the system is contained in the Privacy Map, which truly is not overly complicated and can be reasonably understood by the officers accountable for its decisions.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

5.3.9 Disclosure

As discussed in Section 2.11 – Disclosure requires being able to identify a specific person in the network and this version of Pluto does not support that. It is likely that future versions of Pluto could be extended to allow for network disclosure; that possibility will be discussed in Section 8.4 - Disclosure and the Ability to Identify Individual Users.

This does not however mean that Pluto fails as a Privacy Protocol. However, data servers will need a separate protocol to allow people to identify their own records.

5.3.10 Challenge

The discussion in Section 2.12 – Challenge concluded that there are two requirements that would enable a privacy protocol to allow for reasonable challenges: the ability to describe how privacy related decisions are made, and the ability to change those decisions in a reasonable amount of time.

Both of those requirements are met by the PCC and its Privacy Map. A data server or the privacy officer can describe the system's policy by releasing and explaining the Privacy Map. Further a data server can describe exactly what permissions on the privacy map it participates in and how that relates to its personal information. The Privacy Map provides a great description of how the system handles privacy.

The PCC and its Privacy Map also provide a centralized location to change how privacy is handled in the system. Updating the Privacy Map should generally be a fairly easy operation of adjusting a fairly simple graph, and those changes will then impact all future tickets generated by the system.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Chapter 6

Prototype and Performance Evaluation

6.1 Prototype

Although some of the properties of interest can be directly determined by understanding the protocol, some required testing to ensure reasonable performance. A lightweight prototype was developed in C# for testing; the prototype did much of the significant work and the architecturally-significant functionality such as determining valid permissions, encryption, data flow. Many advanced P3P and Kerberos features however were not fully implemented as they would not affect the contributions of this work which was primarily being done as a proof of concept and not meant for industrial deployment.

The prototype is composed of three separate simple applications: the PCC, a simple client and simple data server. The most complicated component is the PCC which creates the tickets. The PCC design shown in Figure 43:

- **PrivacyControlComponentInterface** – A simple interface class that accepts requests for tickets,
- **TicketFactory** – A factory capable of creating tickets that will be sent through the interface,
- **SessionTicket** - the Pluto session ticket,
- **ClientTicketPortion** - The client portion of the **SessionTicket,**
- **ServerTicketPortion** - The server portion of the **SessionTicket,**
- **EncryptionLibrary** - A library of encryption routines which will be used to encrypt the SessionTicket,
- **PrivacyMap** - The privacy map (as described earlier in 5.2.1.3 Privacy Control Component Setup),
- **Client** - Models clients that are registered with the system,
- **Role** – Models roles in the **PrivacyMap,**
- **Permission** – Models system permissions for the **PrivacyMap**, and
- **Server** – Models servers that have authorized the PCC to create

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

SessionTickets to access its data.

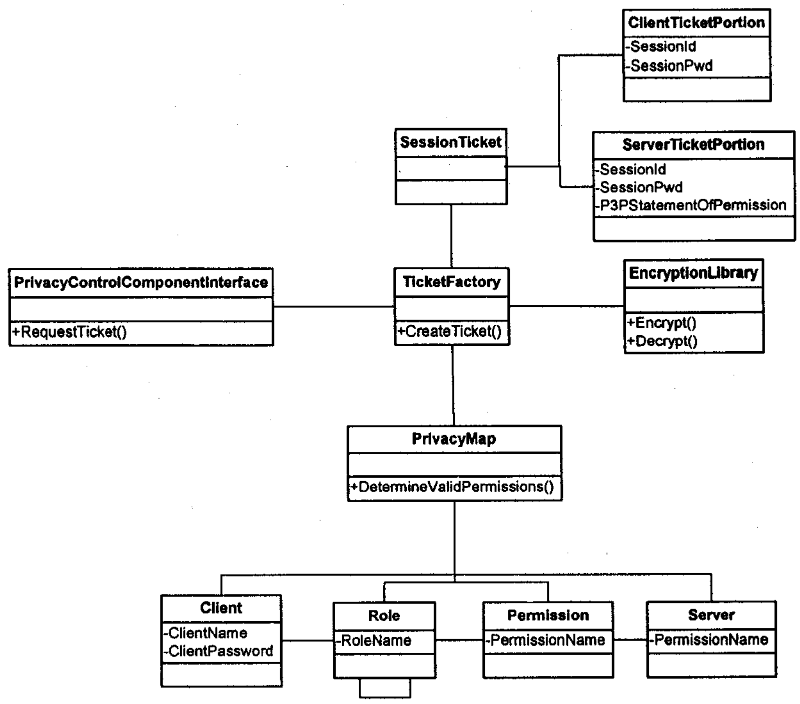

Figure 43 - PCC Design

In order to store models, the Privacy Map can also be stored as an XML file using C# DataSets. In Figure 46 the dataset definitions are illustrated.

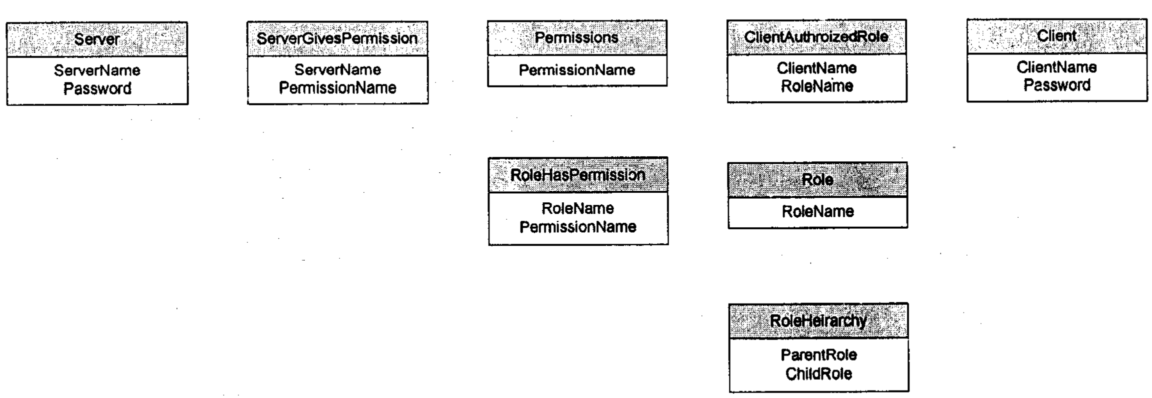

Figure 44 - Privacy Map Data View

The prototype data servers are designed to allow clients to open sessions by providing a ticket and to allow different types of messages to the server. In order to accommodate the required tests, the data server can reply to a simple ping request, to encrypted messages, or to proper Pluto verification requests.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

As shown in Figure 44, the prototype data server application consists of:

- **DataServerInterface** – A simple interface class that allows a client to create a session and make various requests of the data server.
- **DataServerWorker** – A worker class that verifies open sessions and creates responses to client requests that are received by the **DataServerInterface.**
- **SessionTicket** – Similar to the **SessionTicket** in the PCC application except that the **DataServer** does not need to model the client portion of the ticket.
- **EncryptionLibrary** – An library of encryption routines that can be used to decrypt tickets and client requests, this library will also be used to encrypt responses to clients when appropriate.

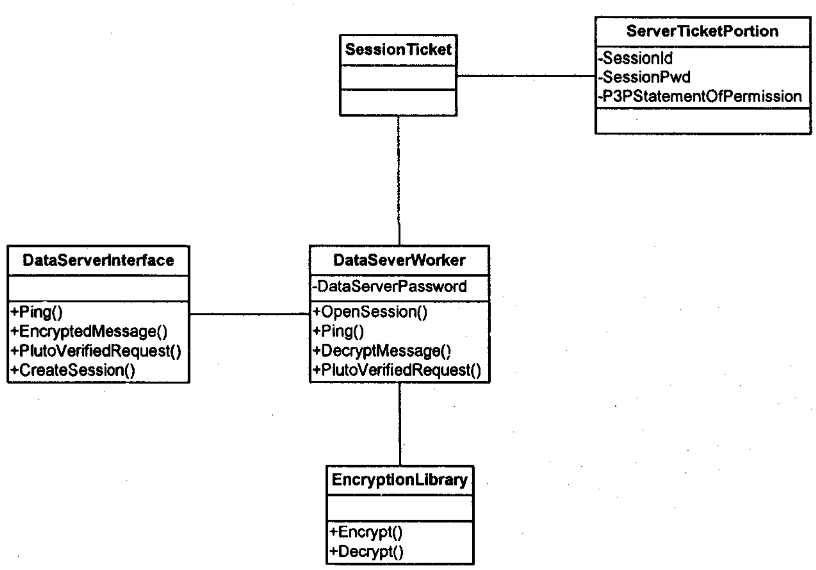

Figure 45 - Data Server Design

The prototype client is a very lightweight GUI application as most of the complex work will be done remotely by the PCC and the DataServer. As shown in Figure 45, the prototype client application consists of:

- **ClientGUI** – A simple GUI that allows users to request tickets from the PCC and to interact with the prototype DataServer.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

- **SessionTicket** – A session ticket similar to the **SessionTicket** in the PCC except that the client does not need to understand or use the server portion of the ticket.
- **EncryptionLibrary** – A library of encryption routines that will be used to decrypt the client portion of the session ticket, and encrypt/decrypt requests and responses from the data server.

The prototype client will need to remotely interact with:

- **PrivacyControlComponentInterface** – The interface to the PCC which the client will need to get new tickets so that it can interact with the data server.
- **DataServerInterface** – The interface for the data server which the client will use to interact with the data server.

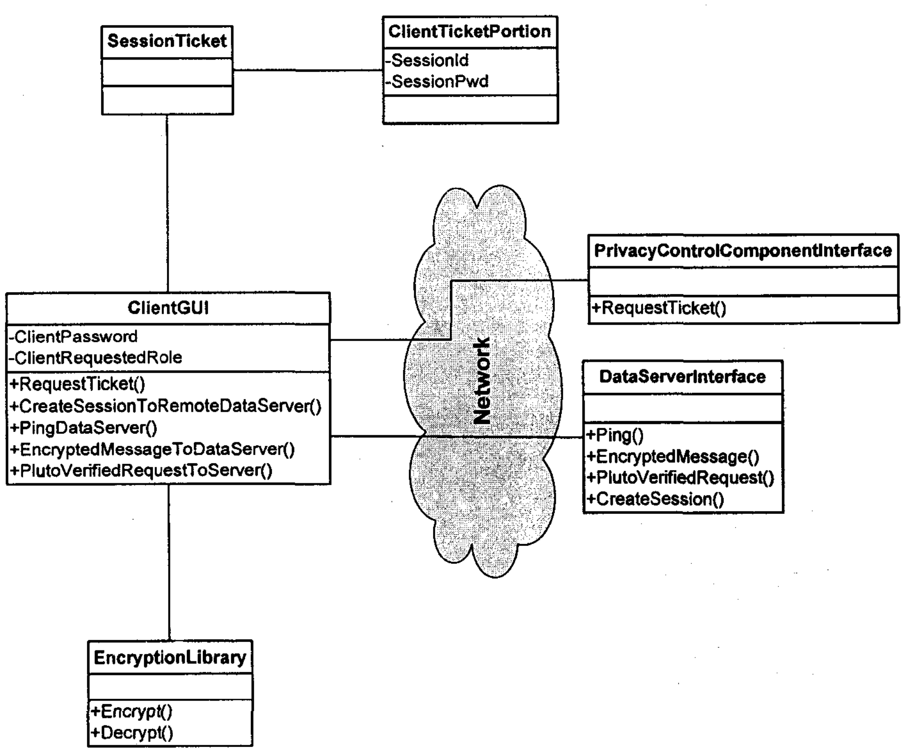

Figure 46 - Client Design

6.2 Evaluation and Results

In Chapter 3, a set of criteria for evaluating possible policy solutions was introduced.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Given the Pluto protocol developed in Chapter 5, it is now possible to evaluate how the actual protocol performs in regard to these criteria. As mentioned earlier, a Session Based Control protocol, Pluto should have the following properties:

Table 2 – Architecture Expectations for Session Based Control Protocols

	Responsiveness	Absence of Data Duplication	Readability	Scalability	Efficiency	Bottleneck Avoidance
Session Based Control	Can be tuned	Good	Good	Can be tuned	Good	Can be tuned

* Note: The above table is very subjective, and is meant to be a simple summary of the more comprehensive discussion below.

All testing computers were Microsoft Windows XP platform with a 2.80GHz Pentium® 4 CPU and 512 MB of RAM. For the second set of tests the machines were connected via a 100 Megabit Ethernet LAN.

6.2.1 Responsiveness

As expected responsiveness can be tuned by using the time to live attribute in generated Pluto tickets. A change by the privacy officer will immediately impact all future generated tickets; however, existing tickets generated with out of date information will still exist. Those tickets are still useable until they have lived longer than their time to live attribute at which point they will no longer be useful.

Pluto responsiveness will therefore be dependent on setting a reasonably low time to live attribute for ticket generation.

6.2.2 Absence of Data Duplication

There is only one real complete system view, and certainly only one view that can be edited: the one in the PCC. The tickets themselves have a copy of some of the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

information in the PCC but, they should not be editable and will expire on their own. Pluto avoids most data duplication problems quite well.

6.2.3 Readability

The PCC contains the bulk of the information required to explain the system to people outside the system. In order for a data server to explain how it might share data with the community it could provide the PCC the privacy map, and explain the permissions that it allows the community to use its data for.

While it might take some assistance to fully explain the system to a person, it is certainly possible. Pluto can create a readable privacy statement.

6.2.4 Scalability

Pluto systems should scale well. After a ticket has been acquired from the server there is no real community interaction; a client with a ticket can talk directly to his desired data server without interacting with any other system resource.

Scalability will be limited by the ability to generate tickets; as there is no large scale system it is difficult to estimate a reasonable rate for ticket generation. However, it is reasonable to assume that in a system where privacy is important that the number of sessions would have some limit. This is not an open network where everyone who can find a way to connect to it might reasonably expect to access data.

In order to understand how many tickets could be generated in a typical system, the test system was used to generate one million tickets. The tickets were generated on a special PCC that generated the tickets locally. This is justified because bandwidth is reasonably cheap and could be added as required for a working model; computation is much more limited. The speed of software to generate tickets is likely an upper bound to the rate at which tickets could be generated; the size of bandwidth is much more malleable and would be less likely to put an upper bound on the creation of tickets in a full

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

implementation.

The ticket generation tests used randomly generated Privacy Policy Graphs. For all these graphs there is a single client who always has access to Role 0, and a single application Server that grants the community all permissions.

The graphs vary over 3 parameters:

1) There are a variable number of roles.
2) There is a variable number of permissions; however all permissions are guaranteed to be reachable from Role 0 (which is useful in some tests most notably the client-application tests).
3) The chance of a Role A being the root of an "is child" relationship to Role B (called connectivity in the following charts).

Figure 47 illustrates these concepts for one possible network.

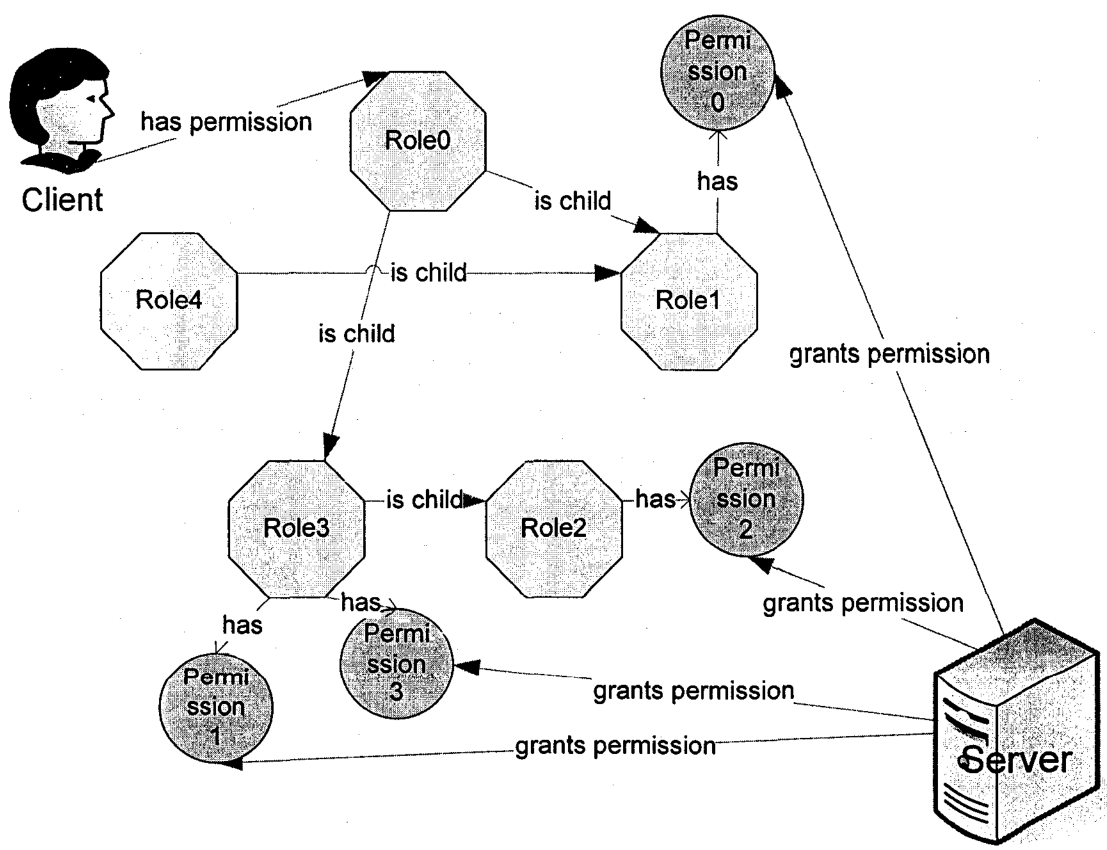

Figure 47 - Possible Test Network with 5 Roles, 5 Permissions and 20% Connectivity

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Two tests were run to show the impact of increasingly large privacy maps:

The first test limited the number of permissions to ten, but used increasingly large numbers of interconnected roles. The expectation is that as the number of roles and their connectedness increased the time to traverse the privacy map would increase and the time to generate tickets would also increase. The results follow in Figure 48.

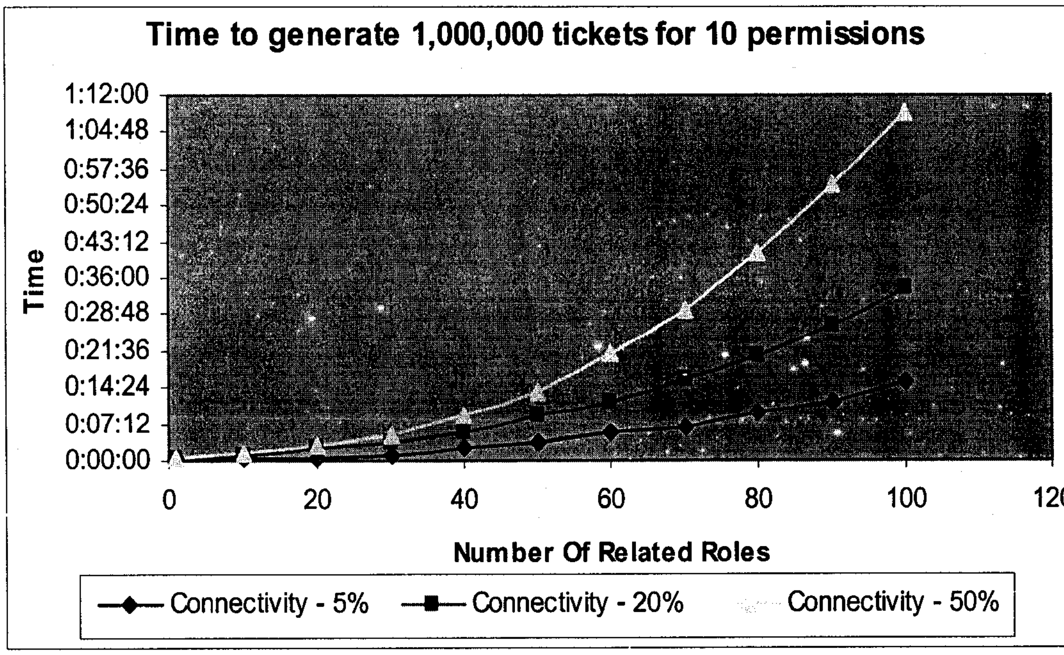

Figure 48 - Time To Generate 1,000,000 Tickets for 10 Permissions.

Even with quite complicated privacy maps including more than 100 heavily connected roles (at 50% connectivity and 100 nodes the test graph would have had approximately 5000 connections to traverse), it took little more than an hour to create one million tickets. As the complexity of the graph decreased, the time to create one million tickets rapidly declined.

The complexity of these graphs does seem to be a reasonable limit for privacy purposes, where a privacy officer would have to completely understand the privacy map, and where the privacy officer would be legally accountable for the decisions made that are based on that graph.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The ability to create roughly one million tickets per hour (or better) also seems reasonable for what must be tightly controlled ticket creation. It is important to remember that these tickets are reusable (until they expire in time); and it is difficult to imagine scenarios where a million clients would be connecting to different unique servers per hour. These results are promising for the likely size and complexity required for a great many privacy controlled systems.

The second test limited the number of roles, while increasing the number of permissions. The expectation was that a large number of roles create increasingly large tickets and increase the time to find the intersection of the requested, client-permitted and server-permitted sets. The results follow in Figure 49.

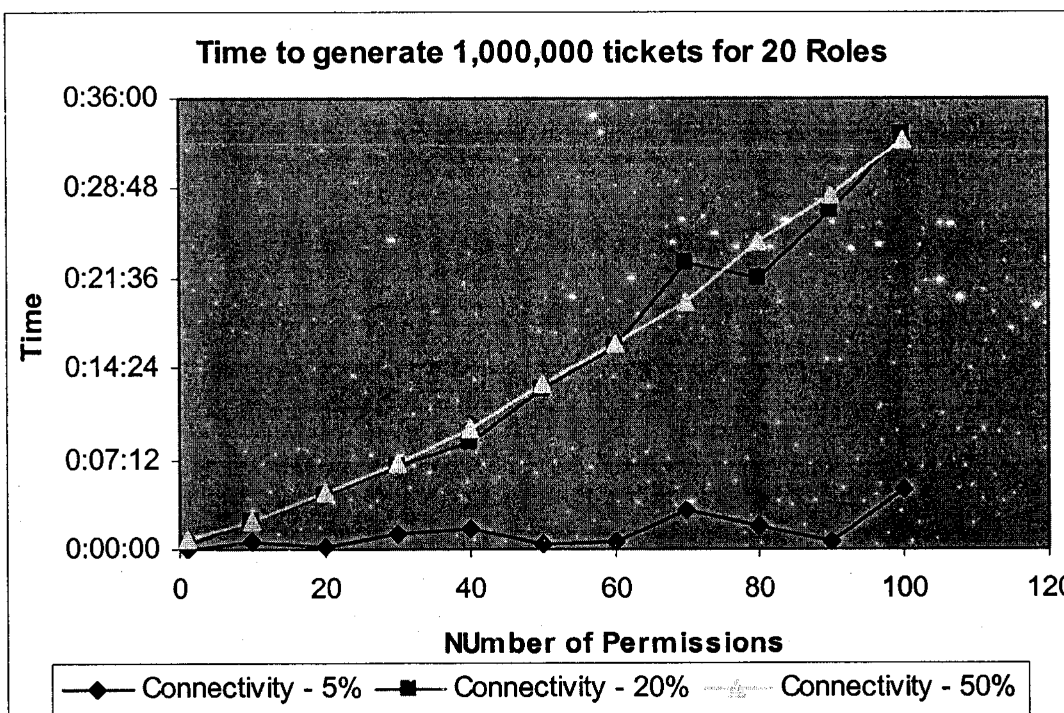

Figure 49 - Time to generate 1,000,000 tickets for 20 roles

The number of permissions in typical systems or even complex systems is likely to be limited to approximately one hundred. Again, a privacy officer is legally responsible for system behavior based on the privacy; and it is unlikely that a person would be comfortable managing privacy maps with more than one hundred permissions. Again, this level of complexity seems to be a reasonable limit for testing.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

As stated earlier, the expectation of increasing time to create tickets held, the 1,000,000 tickets generated in approximately an hour (or less) did seem reasonable for networks likely to require privacy security.

6.2.5 Message Efficiency

A set of tests were run to evaluate the impact of using Pluto Tickets on client/server communication efficiency. Three types of test runs were undertaken to compare the impact of Pluto ticket usage on network connections:

- The first metric was a pure network ping where no meaningful work was done.
- The second test added encryption to the test. A small test message was passed through the application server and encrypted at both ends.
- The final metric was a full Pluto ticket session where all the permissions were ensured before echoing the encrypted message.

The results follow in Figure 50.

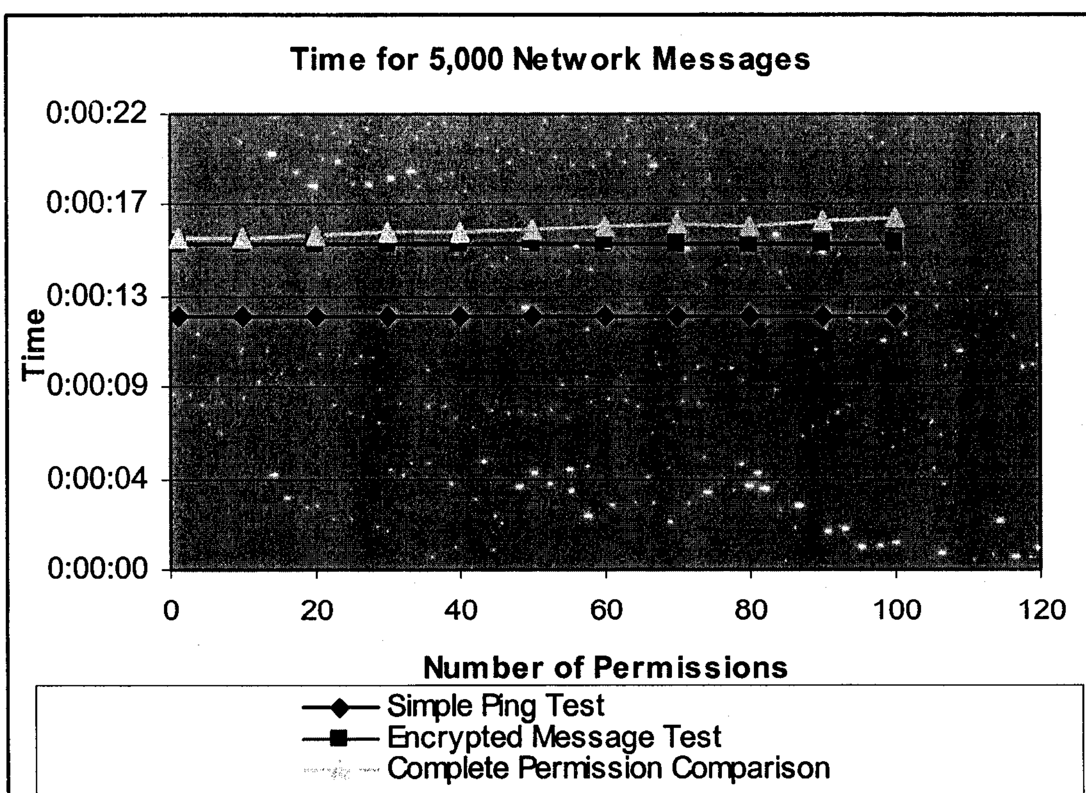

Figure 50 – Time for 5,000 Network Messages

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

It is only reasonable to assume that any system dealing that would pass private information would require both message passing and encrypting of those messages. The interesting question then becomes how much overhead does the rest of Pluto (in particular, permission filtering) likely add to the request/response interaction between a client and its server.

The test results indicate that even if there are a large number of permissions granted to the session, the additional time to fully support the Pluto protocol's permission tests is a very small portion of the time to encrypt and pass the message. Implementing Pluto's permission validation, when compared to implementing to implement any other secure message transmission, should add very little additional overhead.

6.2.6 Bottleneck Avoidance

While there is a bottleneck (the PCC), its impact has been mitigated considerably in the Pluto protocol. The bottleneck is not on data request and response, but on sessions that could (and often will) allow for multiple requests and their responses. As discussed in 6.3.5 it is likely that the protocol is sufficiently powerful for the needs of privacy controlled communities.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Chapter 7

Conclusions and Future Work

This thesis has presented existing privacy legislation and shown how a protocol can be developed to allow electronic communities to share private data in a very controlled way. In particular, the Pluto protocol has been shown to meet the legislated privacy requirements of:

- **Accountability** – by allowing and assisting a privacy officer to run the privacy control policies of a large dynamic electronic community;
- **Purpose Based Storage** – by relying on the principle that data servers understand the reasons they release (and therefore hold) private information;
- **Consent** – by requiring that clients and servers both understand the reason they are sharing data and allowing the server to refuse data requests for purposes beyond those for which it has consent;
- **Collection Limitation** – by restricting how clients can collect data from data servers;
- **Usage Limitation** – by restricting the flow of private information to sessions that were created for valid purposes;
- **Accuracy** – by simplifying the ability to create data; the hope is that it will be easier for clients to get new data from servers than attempt to manage their own data networks which will have the added benefit of increasing the accuracy of data across the community;
- **Safeguards** – by ensuring that no private data is ever transmitted as clear text across the network; the ONLY data that is ever sent as clear text is the client's name.;
- **Open Policy** – by collecting all policy data in a simple and complete form in the privacy control component; and
- **Challenge** – by creating a reasonably simple description privacy information it is possible to to access the policies of a community and by empowering the privacy

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

officer to make changes rapidly that will impact how the community behaves

There is, however, further work required to enable the final parameter of the privacy legislation **(Disclosure).** The primary issue being that the Pluto protocol as it stands does not have any method to identify a person specifically, but only the role or job of a person. The Disclosure privacy policy requires that servers be able to release data about a specific person to that person; but with no way to identify a specific person Pluto can not assist in this one aspect of privacy. This is not a fatal flaw, as a person could still contact a data server using another protocol to get his records and there may be ways to augment Pluto to allow for disclosure, which is left for Future Work).

Prototyping and measurements showed that the proposed Pluto protocol has exhibits most of the desirable properties that a privacy control network would require. Pluto is a Session Based Control solution to privacy control; discussion and testing have shown that it has quite excellent technical properties:

- **Responsiveness** – the PCC can be immediately updated by the privacy officer which impacts all future tickets generated; existing tickets will continue to live for some time but are designed to expire in a timely manner
- **Absence of Data Duplication** – with only a single source for all privacy policy data there is no real data duplication issue
- **Readability** – the privacy policy can be expressed quite reasonably as a set of permissions associated with clients and servers, and the privacy map in the PCC
- **Scalability** – is quite reasonable. There is a limit to the number of sessions that the PCC can generate, but the actual amount of traffic and the number of servers and clients should be very scalable
- **Message Efficiency** – the Pluto protocol has very small overhead beyond the minimal encryption that would be required of any reasonable protocol for passing private data.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The proposed Pluto protocol presented in this thesis lays the ground work for a privacy control solution for electronic communities, but it would require considerable additional work before it could be come a proper standard. This thesis focused on the plausibility and requirements that would be needed for a privacy control standard, and not on the minutia required for a full industry strength standard.

Pluto does not adequately deal with one aspect of the Privacy Legislation: Disclosure. The Disclosure principle requires the holders of data to be able to release information about a particular person to that person.

As it stands, Pluto requires outside support, assuming that clients can contact data servers outside the Pluto protocol to ask for their own records from the data servers. However, it is possible that Pluto could be further amended to identify a particular person as part of its privacy statement. The privacy information could include a specific identifier for the person to whom the ticket was released.

The immediate problem with doing that is that a community might not be able to agree on a standard protocol for describing people. Being able to identify a person in a global marketplace is not a trivial problem as there is no universal identifier: names can be duplicated, and country-specific identifiers (such as Social Insurance Number) may not be available to identify people

This problem is further complicated by privacy as a protocol would likely need to be designed to work with whatever information is available (a person's current SIN number) and not require new special information to be associated with a person. There are likely further legal and ethical issues that would be involved in trying to create a single international standard for identifying individual people. What new privacy problems would be created by an international body that keeps track of individual people?

However, there is work being done on creating universal descriptions of people and objects that could solve identification problem without creating new identification

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

schemes, by working with existing local identification systems. One such effort is the UN/CEFACT's Core Component technology [37].

Further work should probably be done to identify strong candidates for identifying specific people in the Pluto context such that the Disclosure principle can be handled as well. However, much of this work would need to be done at a legal and ethical level before it could be done in a technical system.

Finally, this work was done with the assumption that clients are retrieving information from private servers. However, it is likely that this work could be built upon so that clients could add information to data servers. However, there are definite complexities arising from possible race conditions when two users with similar roles tried to update data at the same time. Further research is required into how Pluto would deal with data update conflicts.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

References

[1] C. Desmarais, X. Shen, S. Shirmohammadi, A. Cameron, N.D. Georganas, I. Kerr, "PLUTO - A Privacy Control Protocol for e-Commerce Communities", Proc. IEEE Conference on Enterprise Computing, E-Commerce and E-Services, Tokyo, Japan, July 2007

[2] C. Desmarais, X. Shen, S. Shirmohammadi, A. Cameron, N.D. Georganas, I. Kerr, "Privacy Management in e-Commerce Communities", Proc. IEEE Canadian Conference on Electrical and Computer Engineering, Vancouver, BC, Canada, April 2007

[3] X. Shen, S. Shirmohammadi, C. Desmarais, N.D. Georganas, I. Kerr, "Enhancing e-Commerce with Intelligent Agents in Collaborative e-Communities", Proc. IEEE Conference on Enterprise Computing, E-Commerce and E-Services, San Francisco, CA, U.S.A, June 2006

[4] X. Shen, S. Shirmohammadi, C. Desmarais, N.D. Georganas, "Multi-agent System Architecture for Collaborative E-Commerce", Proc. IEEE Canadian Conference on Electrical and Computer Engineering, Ottawa, ON, Canada, May 2006

[5] "Personal Information Protection and Electronic Documents Act", S.C.2000, c. 5, <http://laws.justice.gc.ca/en/P-8.6>

[6] Department of Justice Canada, "The National Standard of Canada Entitled Model for the Protection of Personal Information", CAN/CSA-Q830-96 http://laws.justice.gc.ca/en/P-8.6/92379.html

[7] "Fiftieth Anniversary of the UniversalDeclaration of Human Rights 1948-1998, Universal Declaration of Human Rights; Adopted and Proclaimed by General Assembly resolution 217 A (II) of 10 December 1948" <http://www.un.org/Overview/rights.html>

[8] "Personal Information Protection and Electronic Documents Act", <http://en.wikipedia.org/wiki/Personal_Information_Protection_and_Electronic_Documents_Act>

[9] X. Shen, T. Radakrishnan and N.D. Georganas, "vCOM: Electronic Commerce in a Collaborative Virtual World", J. of Electronic Commerce Research and Applications, Vo.1, No.3-4, Aut.-Winter2002, pp. 281-300.

[10] A. Datta, M. Hauswirth and K. Aberer, "Beyond 'web of trust': Enabling P2P E-commerce", Proceedings of the IEEE Conference on Electronic Commerce (CEC'03), June 24-27 2003, Newport Beach, California, USA. pp.303-312.

[11] A. Warms, J. Cothrel, T. Underberg, "Return on Community: Proving the Value of Online Communities in Business", Participate.com, April 12, 2000.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[12] B. Tung, "The Moron's Guide to Kerberos, Version 2.0", 2007, <http://www.isi.edu/~brian/security/kerberos.html>

[13] C. Yang and C. Zhang, "Designing Secure E-Commerce with Role-based Access Control", Proceedings of the IEEE Conference on Electronic Commerce (CEC'03), June 24-27 2003, Newport Beach, California, USA.

[14] C.A. Shoniregun, A. Omoegun, D. Brown-West and O. Logvynovskiy, "Can eCRM and Trust Improve eC Customer Base?", Proc. IEEE International Conference on E-Commerce Technology (CEC'04) pp. 303-310.

[15] D. Ferraiolo, and R. Kuhn, "An Introduction To Role-Based Access Control", NIST/ITL Bulletin, December, 1995 <http://csrc.nist.gov/rbac/NIST-ITL-RBAC-bulletin.html>

[16] D. N. Jutla, E. K. Kelloway, and S. Saifi, "Evaluation of User Intervention Mechanisms for Privacy on SME Online Trust", Proceedings of the IEEE International Conference on E-Commerce Technology, 2004

[17] G. E. Kersten, G. Lo. "Negotiation Support Systems and Software Agents in Business Negoitaions," The First International Conference on Electronic Business, Hong Kong, 2001.

[18]G. Karjoth, M. Schunter, E. Van Herreweghen, M. Waidner, "Amending P3P for Clearer Privacy Promises", Trust and Privacy in Digital Business - TrustBus 03. In 14th Int'l Workshop on Database and Expert Systems Applications (DEXA), Prague, 2003. <http://www.zurich.ibm.com/~mts/bibliography/KSVW_03.P3Ptrustbus03.pdf>

[19] H. Li, D. Ahn, and P. C. K. Hung, "Algorithms for Automated Negotiations and Their Applications in Information Privacy", Proc. IEEE International Conference on E-Commerce Technology, 2004, pp. 255-262.

[20] J. Cole, and C. Neuman, "The Kerberos Network Authentication Service (V5)", Digital Equipment Corporation,
Network Working Group, ISI, September 1993 <ftp://ftp.isi.edu/in-notes/rfc1510.txt>

[21] J. Marathe, "Creating Community Online", Durlacher Research Ltd, 1999.

[22] J.C. Oliveira, X. Shen and N.D. Georganas, "Collaborative Virtual Environments for Industrial Training and e-Commerce", Chapter 7 in "Virtual Reality Technologies for Future Telecommunications Systems", Willey, 2002.

[23] J. G. Steiner, B. Clifford Neuman, and J.I. Schiller. "Kerberos: An Authentication Service for Open Network Systems". In Proceedings of the Winter 1988 Usenix Conference. February, 1988. (Version 4)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[24] I.R. Kerr, "Bots, Babes and the Californication of Commerce", U. of Ottawa Law and Technology Journal,, *1*, 2004.

[25] K. Frikken. M. Atallah and C. Zhang, "Privacy preserving credit checking", Proceedings of the 6th ACM conference on Electronic commerce, Vancouver, BC, Canada, pp.147 – 154, 2005

[26]K. Hornstein, "Kerberos FAQ, v2.0 (last modified 8/18/2000)", United States Government as represented by the Secretary of the Navy, 2000, <http://www.cmf.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html>

[27] L. Guo and N.D. Georganas, "Towards Agent-based Collaborative and Interactive Virtual Environments", Chapter 6 in "Virtual Reality Technologies for Future Telecommunications Systems", Willey, 2002.

[28] L.F. Cranor, Web Privacy with P3P, O'Reilly & Associates Inc., 2002

[29] R. Bond, "The technology works-the law doesn't: legal aspects of secure electronic commerce , eCommerce - Trading But Not As We Know It" (Ref. No. 1998/460), IEE Colloquium on 3 Nov. 1998 Page(s):4/1 – 446

[30] S. Case, N. Azarmi, M. Thint and T. Ohtani, "Enhancing E-Communities with Agent-Based Systems ", IEEE Computer Society, July 2001 (Vol. 34, No. 7) pp. 64-69.

[31] S.Kim and W.W. Smari, "On a collaborative commerce framework and architecture for next generation commerce", Proc. Collaborative Technologoies and Systems 2005

[32] S. Oliveira and O. Zaiane, "Achieving Privacy Preservation When Sharing Data For Clustering, Workshop on Secure Data Management in a Connected World (SDM'04) in conjunction with VLDB'2004", Springer Verlag LNCS 3178, pp 67-82, Toronto, Canada, August 30, 2004.

[33] T. Finin Et al, "KQML as an Agent Communication Language. In The Proceedings of the Third International Conference on Information and Knowledge Management (CIKM)", ACM Press, November 1994.

[34] Y. Lu, W. Wang, D. Xu, and B. Bhargava, "Trust-Based Privacy Preservation for Peer-to-peer Media Streaming" , the 1st NSF/NSA/AFRL workshop on secure knowledge management (SKM'04), Buffalo, NY, Sep. 2004.

[35] Y. Zhang, L. Guo and N.D. Georganas, "AGILE: An Architecture for Agent-Based Collaborative and InteractiveVirtual Learning Environments", Proc. IEEE Globecom 2000 Conference, San Francisco, 2000.

[36] P3P, http://www.w3.org/P3P/

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[37] Keberos, http://web.mit.edu/kerberos/www/

[38] UN/CEFACT Core Components, "UN/CEFACT Core Components Technical Specification. Version 3.0". Second Public Review. 16-April-2007. Copyright © 2007 UN/CEFACT. 123 pages. Previous version: Core Components Technical Specification Version 2.2 (Working Draft B), 31-March-2006.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.