UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Denkleiers • Leading Minds • Dikgopolo tša Dihlalefi

In Search of Search Privacy

by

Wesley Antonio Brandi

Submitted in partial fulfilment
of the requirements for the degree

Doctor of Philosophy

in the

Faculty of Computer Science

of the

University of Pretoria

July, 2011

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

In Search of Search Privacy

by

Wesley Antonio Brandi

Supervisor: Prof. Martin S Olivier
Department: Computer Science
Degree: Doctor of Philosophy

Summary

Search engines have become integral to the way in which we use the Web of today. Not only are they an important real time source of links to relevant information, but they also serve as a starting point to the Web. A veritable treasure trove of the latest news, satellite images, directions from anywhere to anywhere, local traffic updates and global trends ranging from the spread of influenza to which celebrity happens to be the most popular at a particular time.

The more popular search engines are collecting incredible amounts of information. In addition to indexing significant portions of the Web they record what hundreds of millions of users around the world are searching for. As more people use a particular search engine, it has the potential to record more information on what is deemed relevant (and in doing so provide better relevance in the future, thereby attracting more users). Unfortunately, the relevance derived from this cycle between the search user and the search engine comes at a cost: privacy. In this work, we take an in depth look at what privacy means within the context of search. We discuss why it is that the search engine must be considered a threat to search privacy. We then investigate potential solutions and eventually propose our own in a bid to enhance search privacy.

Acknowledgements

My thanks to everybody who stood by me through the years that it took to get this done. Prof. Olivier for believing in me way back in the day and providing subtle hints and tips in a way only he knows how. Olman and olady for their unwavering support. My sisters for putting up with me in general and my brother for the bizarre ideas and crazy conversations.

The whole experience: full time, part time, on campus, off campus, jobs, sundays, bursaries, friends, labs, exams, tutorials, workgroups, papers, reviews, conferences, questions, rejections, acceptance, and on and on. All of it an incredible journey that I cherish and remember fondly.

Happy days.

Contents

List of Figures

List of Tables

Chapter 1

Aims and Scope

"Everything counts in large amounts"
Depeche Mode

1.1 Introduction

Finding information on the Web of today requires that one need only know the address of a popular search engine. With several keystrokes and the click of a mouse, a user can issue a query to one of the largest information repositories on the planet, only to have it processed and a response returned in what is often less than a second. Users of the Web have learned to expect nothing less from such services. Not only are they expected to be free, but there is incredible pressure amongst the bigger search engines to crawl as much of the Web as often as they can so as to be in a position to provide their patrons with the most relevant results possible (or at least better than the results offered by their competitors).

Maintaining a search engine that can scale with the complexity and vastness of a network the likes of the Web is not trivial. Indexing the Web requires that significant resources are made available to the search engine. At the very least, an effective search engine must have an army of crawlers (these are programs that index Web pages) which are constantly exploring new content in addition to updating their existing content. The cost and complexity of maintaining a system of this nature is such that there are only a few search engines in the world that can afford to be competitive. As a result, hundreds of billions of search queries belonging to hundreds of millions of online users are essentially funneling down to an extraordinarily small set of search engines (in the United States there are only five search engines of significance[1]).

The threat to privacy in this scenario is dire. On numerous occassions, privacy experts have highlighted these threats and warned of the potential for a privacy nightmare.

The more popular search engines have responded to these concerns with mitigations that are not apt. When perusing the privacy policies of two of the bigger search engines of today (Bing[2] and Google[3]), the mitigations provided in order to protect one's privacy (anonymising one's IP address and browser cookies after a period of time) contribute very little, if at all. Despite the legitimate concerns of the privacy community, the search engines have not engaged in a manner that is satisfactory. It seems that some do not even appreciate the gravity of the problem. Consider the following statement on privacy from Eric Schmidt, the CEO of the biggest search engine on the Web today (Google) in an interview with CNBC [20]: "If you have something that you don't want anyone to know, maybe you shouldn't be doing it in the first

[1] comScore Releases February 2009 U.S. Search Engine Rankings, http://wa.la/3E2

[2] Bing Privacy Supplement - http://privacy.microsoft.com/en-us/bing.mspx

[3] Google Privacy Center - http://www.google.com/privacy_faq.html

place."
Obviously, this statement is incredibly short-sighted. Whilst it may apply to some who are engaging in illegal activities and want it to be kept a secret, it does not apply to the individuals who have legitimate reasons for maintaining their privacy (those investigating antiretroviral drugs, abortion clinics, abuse et cetera).

The lack of commitment to privacy from the search engines has not gone unnoticed by the privacy community. There are a number of Privacy Enhancing Technologies (PETs) on the Web that aim to enhance and protect one's privacy. These include tools that are the result of projects conducted at universities as well as products shipped from multinational software corporations (the privacy options in Microsoft's Internet Explorer, for example).

In this work, we take a detailed look at privacy within the context of search. Given that there are a number of PETs which contribute to online privacy already, we ask if this is sufficient to protect one's search privacy. As we will discover, this is not the case. Despite the PETs that are available, when one considers the nature of search and what privacy in this context means, there is still work to be done. Upon identifying the search engines as entities that cannot be entrusted with one's search privacy, the work required can not come from them.

As a result, we will investigate potential solutions to the search privacy problem that do not rely on the cooperation of the search engines.

This work will contribute the following:

- We look at each of the Privacy Enhancing Technology (PET) categories and try to find an existing solution to the search privacy problem, i.e., using existing technology, can the search engine be eliminated as a threat to search privacy?

- Having found that the search engine remains a threat as long the original use case scenario for search is maintained, we confirm that users are sharing search queries in a manner that facilitates a network acting as a cache of search queries and results. We describe this network and discuss its contribution to preserving the search privacy of its users.

1.2 Outline

As mentioned above, the primary contribution of this work is the proposal of a search network. We build up to this by evaluating each PET category in terms of what it can deliver to search privacy. If the search engine can be eliminated as a threat to privacy using existing means, then the problem is solved. Of course, as we will see, this is not the case.

This document is structured as follows:

- Chapter 2 begins with a discussion of the definitions available for privacy. We don't try to redefine or provide our own definition in this chapter. Instead we arrive at an existing and applicable definition given the nature of the Web. We then discuss privacy of one's identity (anonymity) and move on to look at each category of PET available today.

- In chapter 3 we discuss search in more detail. Specifically, we look at its evolution through time and analyse the three key contributors to the search scenario of today: search users, search engines and crawlers. This leads to a discussion of search profiles, both online (the Web) and offline (the library). After highlighting a search privacy violation via AOL in 2006, we conclude this chapter by identifying which categories of PETs may be applicable in addressing the search privacy problem.

- Anonymity is the first category which showed promise in addressing the problem. In chapter 4 we look at this category and describe an attack against crowds based upon two principles, that of unlikely senders and recognising related requests. We then present and discuss the results of a simulated attack.

- We turn to the category of personal control in chapter 5. The Platform for Privacy Preferences (P3P) embodies the definition of this category and is examined in detail. We highlight two problems with this technology, that of trust and proxies, and investigate possible solutions.

- Private communication through encryption and obfuscation is examined in chapter 6. We take a closer look at technology which uses obfuscation in a bid to preserve one's privacy. We then discuss several problems with this approach and present countermeasures that can be taken by the search engine.

- We begin chapter 7 with an analysis of the database released by AOL in 2006. We use this to confirm that users are sharing a significant amount of search queries. We then leverage off of this fact and propose a search network built upon a Distributed Hash Table with the primary objective of preserving the search privacy of its users.

- This work is concluded in chapter 8. In light of the contributions made, we ask if search privacy has been enhanced and whether or not the search engine is still a threat to search privacy.

Chapter 2

Privacy Enhancing Technologies

Privatus - not in public life, past participle of privare, to release, deprive, from privus, single, alone

2.1 Introduction

In this chapter we examine several technologies that allow users to control their information on the Web; these are typically referred to as Privacy Enhancing Technologies (PETs). PETs range from simple standalone email delivery solutions to complex systems built upon multiple layers of cryptography and implemented across servers around the world.

Before presenting an analysis of technologies that preserve our privacy, we ask: what is privacy? This question may seem simple, but the answer is by no means trivial. In fact, for a term that almost everybody recognises immediately, it is somewhat surprising to note that there is no globally accepted definition [36, 120, 79].

The first goal of this chapter is not to answer this question ourselves, i.e., we will not offer our own definition of privacy. Instead, in the next section we discuss previous research in the field of privacy and look at some of the definitions that have been offered over time. This leads us to a definition of privacy that suits our needs when discussing privacy in the context of PETs on the Web of today.

Having made it clear what is meant by the term privacy when speaking of Privacy Enhancing Technologies, we then provide a brief discussion of anonymity followed by an analysis of some existing PETs.

2.2 Privacy

One of the earliest publications on privacy is an argument for the right to it by Warren and Brandeis [119]. They discuss the broadening of legal rights and equate the right to life with their definition of privacy: “the right to be let alone”. Written during the rise of the printing press and the inevitable mass distribution of photographs, the authors emphasise that under common law, each individual may determine “to what extent his thoughts, sentiments, and emotions shall be communicated to others”.

Introna et al [67] argue that this definition is far too restrictive. They discuss two counter examples: (1) if one were to use a telescope to observe your movements from a distance, this would in fact be leaving you alone, but is obviously not respecting your privacy and (2) there are certain entities/individuals that have a right not to leave you alone, for example, the tax service or your creditors. The authors comment that although privacy, or the absence thereof, is generally easy to identify, it is difficult to define. Even when given a definition, a counterexample can always be found. In arguing that conflicts between different stakeholders of a privacy claim may result in different conceptualisations of privacy, Introna et al describe the notion of privacy as "the freedom from the judgement of others".

Solove [112] argues that "the right to be let alone" definition of privacy is far too broad a definition of privacy. In deconstructing the "nothing to hide" argument against privacy, Solove discusses the difficulties inherent in attempting to find a common core in privacy: if a conception of privacy is too broad, it may encompass too much and become unusable, alternatively if it is too narrow, it may be too restrictive and suffer the same fate. Solove proposes that instead of trying to conceptualise privacy using traditional methods (finding a common denominator), we must instead "understand privacy as a set of family resemblances" [111]. Furthermore, Solove notes that "privacy is not reducible to a singular essence; it is a plurality of different things that do not share one element in common but that nevertheless bear a resemblance to each other."

Boyle [24] notes that privacy is an overwhelmingly large and nebulous concept. Having reviewed existing privacy literature, he still found himself confused as researchers used the term "privacy" to refer to different concepts. DeCew [36] notes that discussing the concept of privacy is difficult because on the one hand it appears we value it as a means to provide us with a sphere within which we can operate free from the interference from others, yet on the other hand this can be exploited to hide domination, degradation or physical harm.

McParland et al [84] point out that conceptual and operational confusion around privacy has resulted in researchers narrowing down definitions of privacy to fit their field of interest in a bid to avoid some of the confusion and, as a result, the confusion surrounding privacy remains undiminished.

Bellotti and Sellen [116] point out that any definition of privacy must be dynamic. They then define privacy as a subjective notion which may be influenced by culturally determined expectations and perceptions of one's environment. Gavinson [55] describes privacy as the protection from being brought to the attention of others and notes that upon being brought to the attention of others, this will ultimately result in a further violation of privacy (due to further scrutiny).

Westin's definition [120] (arguably one of the most popular) places emphasis on control over an individual's information and describes privacy as "the claim of individuals, groups or institutions to determine for themselves when, how and to what extent information about them is communicated to others." This definition has lead to the widely used term "informational self-determination" and, as Ng-Kruelle et al [86] point out, is widely used in privacy related legislations.

In an argument against the right to privacy, Thomson [115] points out that one can lose *control* over one's information and yet still maintain one's privacy. She discusses an example where a neighbour has an X-ray device which has the ability to see through walls. If you were to come home and close your doors, the fact that the X-ray device exists next door means you no longer have control over your privacy, but your privacy is not violated until the device is used. She goes on to suggest that the right to privacy is itself a cluster of rights and that before we mark off something as a violation of the right to privacy, we should examine whether or not it is the right to something else that has been violated (or perhaps no violation has occurred at all).

2.2.1 Privacy in this research

The Web is founded upon a request/response protocol: a client makes a request to a Web server, this is processed and a response is then provided. Requests to a Web server can vary tremendously: sending/receiving email, shopping, viewing the news, watching a movie, looking up football scores, commenting on a paper or talking to a friend for example. Regardless of the activity, the smallest interactions on the Web result in the ability to store information related to each request. The most basic type of information includes the source from which a request originated (the IP address). Although an IP address only identifies a source of a query for a particular request at a specific time, it can be deemed to be information owned by an individual in the case where the IP identifies the individual (in the event of statically assigned IPs for example).

Consider the case of an individual who considers it a privacy violation to track his browsing habits across a particular site. Storing which pages were requested at what time, and by which IP would in effect compromise this user's privacy because the server would be indirectly tracking him. In this scenario if the user wanted to browse in private he would have to have a say over (or at least be aware of) which of his information is being stored and to what extent, i.e., he would be exercising control over his own information. By him not having control over his information, the server would keep logging his IP and he would not have the benefit of privacy.

As has been pointed out, the notion of control over one's information is emphasised in Westin's definition of privacy. Within the context of the Web and PETs, this can be succinctly captured as *the ability of the individual to protect information about himself* [58].

2.3 Anonymity

Privacy is not anonymity. Within the context of the privacy definition we have chosen to follow, anonymity is a form of privacy in the sense that one has control over one's identity [58]. Activities conducted in an anonymous fashion may still result in a violation of privacy.

Pfitzmann and Koehntopp [94] define anonymity as "the state of being not identifiable within a set of subjects"

Various papers [80, 58] discuss the positive and negative aspects of anonymity on the Internet. In favour of anonymity is the extended support for members of support groups (rape victims, recovering alcoholics and others), whistleblowing, refereeing for academic conferences, anonymous tips to investigative journalists et cetera. Disadvantages of anonymity include exploiting email services to spam the masses, launching massive denial of service attacks and illegally distributed copyrighted software.

Goldberg et al [58] divides anonymity into two categories:

> Persistent anonymity - this has a user making use of a pseudonym which can be used to link a number of transactions to each other, but not to the person behind the pseudonym. This may include communicating via email under a different identity.
>
> One-time anonymity - refers to independent transactions that cannot be linked together. An example may include commenting on blogs through an anonymisation service under randomly chosen names.

Pfitzmann et al [95] discuss exactly what forms of anonymity are possible on a network:

> Receiver anonymity - referring to the receiver of a message. The sender may be known, the message itself may be observed but the receiver of the message is anonymous.
>
> Sender anonymity - referring to the entity from which the message originated. This is the same as receiver anonymity except it applies to the sender only.
>
> Unlinkability of Receiver and Sender - this form of anonymity hides the relation between the sender and the receiver of a message.

There are a number of anonymity models that provide anonymity in one form or another. Of the most popular is that of the mix machines. Chaum

[26] employs the use of mix machines to delay the delivery of encrypted messages so as to hide the source of the message (typically used for email).

The mix approach does not require a trusted central authority. In fact, it is assumed that anyone in the system may learn the source, destination and representation of all messages in addition to having the ability to insert, modify or remove messages. The mix system has its roots in public key cryptography and begins with a user attaching a random string of bits to the message being sent and encrypting this with the receiver's public key. This is then combined with the receiver's address and another random string of bits, encrypted with the mix machine's public key and delivered to the mix machine.

The aim of the mix machine is to deliver the message to B in a manner that does not reveal that it came from A. Since anyone on the mix network can observe the source and destination of all messages, the mix machine cannot simply forward the encrypted message to B since any observer would note that a message from A went to the mix, and onto B. As a result, A must be communicating with B. Instead the mix machine only outputs messages in batches where each message is uniform in length and ordered in a manner that cannot be predicted. While a single mix machine is thought to be ideal, in practice a number of mix machines feed into one another to make for mix cascades. This cascade of machines makes it significantly more complex to perform traffic analysis. Wright et al [122] provides an overview on a host of different mix implementations.

2.4 Privacy Enhancing Technologies

In this section, we discuss a number of PETs on the Web today. Each PET is categorised into one of the key PET categories identified by Olivier [89, 91]:

Private communication - this category consists of technologies that provide the ability to communicate content only to the recipient(s) specified, regardless of who is listening.

Anonymity - these technologies are either based on Chaum's mix or implemented in one way or another through the notion of a proxy.

Personal control - defined as "the use of technology to ensure that an individuals personal information is only used in a manner commensurate with the individuals privacy policy."

Organisational Safeguards - similar to personal control, but focused on the organisation: "the use of technology to ensure that the organisation

> complies with its own privacy policy as well as the preferences of the individual".

There are a number of papers that provide an overview of PETs [64, 58, 56, 57, 82, 109]. Each of the technologies discussed falls into at least one of the categories listed above.

2.4.1 Private Communication

Olivier [89] highlights private communication as fundamental to the other categories since the absence of it would weaken almost all of the remaining solutions. For example, consider the case of sending your medical details online to your doctor. When these details are not encrypted, the fact that your doctor has privacy controls in place to protect your data and your privacy is of little consequence if your Internet Service Provider is logging all content processed through it (and in turn logging your medical data).

Goldberg [57] highlights the following private communication technologies:

> Pretty Good Privacy (PGP) [108] - widely available software that leverages off of public-key cryptography [103]. It is popular in email because it allows a sender to easily encrypt/sign the contents of an email in a manner that guarantees: (1) only the intended recipient(s) can read the email, (2) the integrity of the message is preserved (encrypted or not) and (3) non-repudiation from the sender's side, i.e., only the sender could have written the message.
>
> Secure Sockets Layer (SSL) [51, 118] - a predecessor to Transport Layer Security (TLS) [40, 41]. Since there are a number of unknown entities that may be involved in delivering messages from a Web server to a client (routers/transparent proxies et cetera), SSL and TLS are communication protocols supported by most popular browsers that guarantee the integrity of messages between applications on a network in addition to encrypting its contents and authenticating the parties involved (typically only the Web server is authenticated).
>
> Off-the-Record Messaging (OTR) [23] - in contrast to PGP, OTR is a communications protocol which uses encryption to provide repudiation, forgeability and perfect forward secrecy [61]. Perfect forward secrecy guarantees that the key used for a session cannot be learned if the public/private key set from which it was derived is compromised. Repudiation is provided because messages between parties are never

> proved to be from any single party (digital signatures are only used in creating a session). Forgeability is delivered in the form of malleable encryption, i.e., ciphertext can be modified to make meaningful changes to the plaintext even if the key used to generate the ciphertext is not known. These properties ensure that once a conversation is over, nobody can verify the authenticity of any of the messages and since the messages could have been forged, the participants are guaranteed confidentiality. An example of where these properties would be of interest is the scenario where journalists are communicating with informants.

Olivier [89] notes that steganography is receiving renewed interest as a private communication technology. In a paper clarifying the field of steganography, and discussing the limits thereof, Anderson and Petitcolas [9] point out that steganography is not to be confused with cryptography. The latter transforms a message so that it is indecipherable to those who intercept it whereas the former deals with hiding the very existence of the message so that it is only known to the intended recipients. A simple example of steganography includes setting the least significant bits of image pixels to the bits of the message one wants to hide [117].

Howe et al [66] focus on the privacy of a user's search profile and introduce the TrackMeNot (TMN) extension to the Firefox Web Browser. Essentially, TMN assumes that the organisation that will be receiving and storing a user's search queries cannot be trusted. As a result, the aim of this technology is to thwart the logging activities of major search engines through obfuscation. This is achieved by issuing a number of false search queries over time on the user's behalf. In doing so, the real search queries issued by the user would form a small subset of the total queries issued, supposedly leaving the search profile unusable in so far as applying to the user it represents.

TMN has no third party dependencies and uses dynamic query lists to generate queries that are unique to the user in question. The component starts with a dictionary of random terms, seeded from a public source that is then used to issue queries. The search result of each query is analysed and a subset of terms is non deterministically extracted and used to replace terms already in the dictionary. As a result, the terms used and queries generated become unique to the user in question, in addition to looking very much like a real user that is searching. Although the statistical significance has not been proven, at the very least this extension generates a significant amount of noise in a manner that would make extracting the genuine queries (those explicitly issued by the user) cumbersome.

2.4.2 Anonymity

Federrath [47] illustrates the development of modern PETs through a timeline which begins with two important contributions:

- 1978, Public-key encryption. A technology that has already featured in the private communication category, this form of encryption is an important building block in anonymising technologies.
- 1981, Chaum's mix. Discussed in a previous section, most anonymity models are based either on the mix or the notion of a proxy.

Since Chaum first proposed the mix, there have been a number of anonymisation technologies modelled on top of it and the addition of a proxy [99, 42, 100, 80, 33, 1, 69, 100, 90]. The technologies discussed in this section form a small and well known subset of this literature. As a result, we only discuss core concepts and provide relevant references to the reader in the event that more detailed information is required.

Anonymising proxies

Levine and Shields [80] define a proxy as a single server that accepts connections from an *initiator* of an anonymous connection (party A) and forwards them on to the *responder* (party B). The assumption made when dealing with an anonymising proxy is that it will deliver messages from party A to party B but will not disclose to party B that party A is the source, i.e., party B will see the anonymising proxy as the source (this is sender anonymity, as discussed in section 2.3).

Of course, an implicit assumption is that party A trusts the anonymising proxy since although A is anonymous to B, he/she is not anonymous to the anonymising proxy. The Lucent Personalized Web Assistant (LPWA[1]) [52] and Anonymizer.com[2] are technologies built upon this premise. Anonymizer.com is simple, easy to use and, as Goldberg [57] remarks, is one of the few commercially successful anonymity technology providers. Anonymous-proxies.com[3] on the other hand, maintains a private database of anonymous proxies on the Web which is accessible upon paying a monthly fee.

[1] LPWA, available online at http://www.bell-labs.com/projects/lpwa

[2] Anonymizer - Anonymous Proxy, Anonymous Surfing & Anti Spyware, available online at http://www.anonymizer.com

[3] Available online at http://anonymous-proxies.com/index.html

Crowds

Much like anonymising proxies, crowds [100] offers various degrees of sender anonymity through the notion of blending into a crowd. The degree of anonymity, with respect to the sender of a message, is defined as an informal continuum ranging from *absolute privacy* to *probable innocence* and eventually *provably exposed* (refer to [39] for more detail on quantifying anonymity).

Each member in a crowd (a jondo) receives and forwards messages on behalf of any other member in the crowd, i.e., messages are proxied from one jondo to another. When a jondo decides to send its first message via the crowd, a random path must be configured to serve as the route taken for all messages sent by that jondo from that point onwards. The route is established as follows:

1. The sender picks a random jondo and forwards the message to him.
2. This jondo then flips a coin which determines whether to forward the message to yet another jondo (in which case this step is repeated) or to send the message to the intended recipient.

Upon receiving a response from the intended recipient, the message is routed back along the path to the initial sender of the message.

The authors of crowds define three types of attackers:

- A local eavesdropper is defined as someone that can monitor all communication from a single computer (jondo). Though no sender anonymity is offered from this type of attack (since the attacker can see all messages initiated from the computer), Reiter and Ruben describe how receiver anonymity (the end server) tends towards *beyond suspicion* as the size of the crowd tends towards infinity.
- Collaborating crowd members are seen as jondos that pool their information regarding the crowd together in an effort to expose crowd members. An attack is described where collaborators on a path of jondos used to send a message try to determine if their immediate predecessor on the path is the sender. Reiter and Rubin conclude that *probable innocence* is guaranteed for sender anonymity as long as $n >= 3(c+1)$ where n is the size of the crowd and c the number of collaborating jondos. Wright et al [124, 123] formally define this type of attack (the *predecessor attack*) and examine it within a broader scope. In [125], Wright et al formally prove that the predecessor attack is successful against all existing anonymous protocols.

- The end server: Reiter and Rubin make a strong case for sender anonymity against an attack from the end server. They suggest that the anonymity of the initial sender of the message is *beyond suspicion.*

Onion routing

Onion routing delivers private communication within the context of a public network [99, 59]. Sender anonymity is provided in the form of a dynamic path of mix machines between the sender and recipient. A data structure representing an onion is made up of a number of encrypted layers and constructed by the initiator of the message. Each layer of the onion represents a mix machine that will form part of the path that the message will take from the sender to the recipient. As the onion makes its way through the network, each router on the path extracts a layer away in order to determine where to send it next. Encryption at each layer ensures that no router on the path can know the full path that the message will take.

Tor[4] is an example of an anonymising network that is based on second generation onion routing [42]. In 2006, it was estimated that Tor had approximately 450 server nodes participating in its network [92], although as of November 2008 Tor Status services[5] pointed to as many as 1,239 nodes on the network.

Essentially, Tor provides a high degree of unlinkability between the sender and receiver of a message even in the case of compromised mixes. Note that Tor is susceptible to predecessor and intersection attacks. In an intersection attack, the attacker repeatedly tracks which nodes are active through time in an effort to determine who is communicating with which recipients. By exploiting the fact that onion routers may fail or leave the network, an attacker knows that any path of communication that is still functional after a node has left, does not include this node. Similarly, new additions to the network will not be active in any existing paths. Berthold et al [22] provide a comprehensive list of the types of attacks that are possible against mix-based networks the likes of Tor.

2.4.3 Personal Control

We highlight several technologies that provide personal control using separate methodologies: P3P, IE8 InPrivate Blocking and Cookie managers.

[4] An anonymous Internet communication system - http://tor.eff.org/

[5] Available online at http://torstatus.blutmagie.de/

P3P

The Platform for Privacy Preferences (P3P) framework allows Web users to automate the protection of their privacy. Web sites publish P3P policies clearly describing what they intend to do with the data they collect. Web users then compare these policies to their own set of privacy preferences. Users can express their privacy preferences either through the APPEL preference language [78] or through the XPath-based preference language [6]. Provided that the P3P policy published by the Web site is acceptable (they align with the user's privacy preferences), the user may browse the site without interruption.

P3P has been the focus of much research and criticism [30, 29, 32, 65, 114, 104]. Common criticism of P3P includes that it is too hard to use and that there is no means of enforcement, i.e., a Web site with suspicious privacy practices has no incentive to declare what it is doing. Unless legislation codifies privacy protection of this nature, sites with bad privacy practices can simply mimic the policies of sites with good practices. With this in mind, the World Wide Web Consortium (W3C) describes P3P as a technology that is "*complementary to laws and self-regulatory programs that can provide enforcement mechanisms*" [31].

IE8 InPrivate Blocking

The Web has been designed in a manner such that navigation to a single URL may result in fetching content from a number of disparate sources. For example, loading the Wall Street Journal[6] in a browser will result in content delivered directly from the Wall Street Journal's Web servers in addition to references to at least two separate sources for advertisements. Typically, the browser loading the site will then make requests to each of the ad servers for the content in question.

Although primarily not an advertisement blocker, the IE8[7] InPrivate Blocking feature scans for third party sources that are referred to a predefined number of times during the course of navigation through the Web (advertisements typically fall into this category). If a single source exceeds this threshold, then it will be marked as having the ability to track a user's surfing habits (by logging his/her IP) and, as a result, although Web sites may still refer to them for content the browser will no longer request content from these servers. Figure 2.1 illustrates this process.

[6] Available online at http://online.wsj.com

[7] IE8, available online at http://www.microsoft.com/ie8

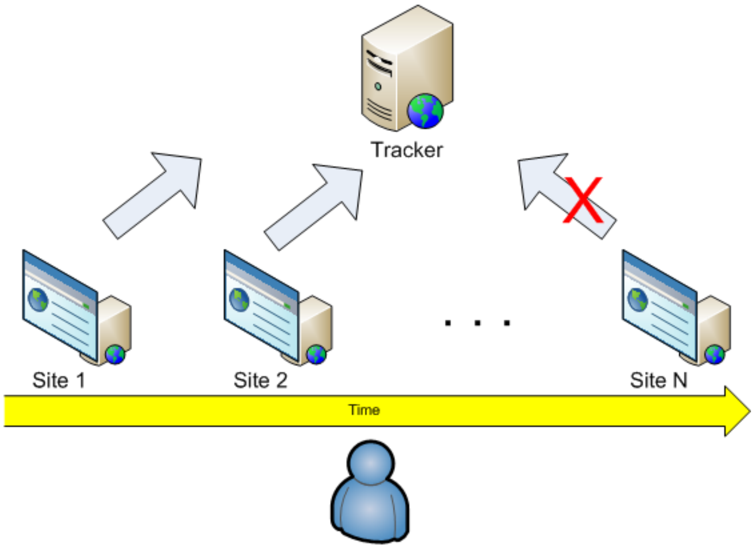

Figure 2.1: A user browses the Web and navigates to several different sites. Each of these sites refer to the same third party Web server. This could be for third party content, counters, advertisements et cetera. In this example, the IE8 InPrivate Blocking feature is enabled and has a threshold of N. On the Nth occurrence of referring to the same third party site from a number of different sites, the feature marks the third party site as being in a position to track the user in question. Subsequent requests to this third party are therefore no longer addressed, i.e., when a Web site refers to this server for content, IE8 will not retrieve it.

Cookie managers

Cookie managers grant control to the user in so far as determining which cookies are accepted from a Web server and how long they will be valid for. Cookies are a means for Web servers to persist small bits of information to a user's machine via the Web browser. This information can be used to enrich the Web experience. Examples may include setting a cookie that is a unique identifier which links to a shopping cart of items on the Web server. A user returning to a Web site need not restock the shopping cart since the Web server will be able to match the user to a cart through the unique identifier.

In analysing different types of PETs, Hall [64] describes cookies as neither good nor bad. Much of the criticism surrounding cookies is targeted at the nature in which they can be persisted, i.e., they can be written to disk without the user's knowledge or approval.

Cookie managers are built into most modern day Web Browsers and can be configured to include the following:

- deny all cookies
- prompt the user to accept/deny all cookies
- accept certain cookies by default
- clear all cookies upon closing the browser

2.4.4 Organisational Safeguards

Organisational safeguards serve as a means to ensure that organisations adhere to the commitments made in their privacy policies.

Consider the Internal Revenue Service (IRS) as an example. To safeguard the tax returns of tax payers, tax assessors need not be granted access to all tax returns all of the time. The organisation could implement mechanisms that only allow tax returns to be assessed by assigned individuals and only for the period for which an assessment is valid. Of course, there are exceptions (in the event of an audit, for example). Furthermore, in so far as issuing tax refunds to tax payers, the department responsible may only require access to their bank details and not the tax return in question. As a result, the IRS could store the bank details in one database and the tax returns in another, with separate DBAs assigned to each.

Olivier [91] highlights a common reservation with any of the technologies that fall into this category: the doubt that businesses, unless forced, will implement them at all.

E-P3P

Using the Web, organisations can declare their privacy policy in a machine readable fashion (via P3P files) or in a separate section of their Web site (typically in natural language). The privacy policy of an organisation essentially states which data will be collected from the individual and what the organisation intends to do with it.

The presence of a privacy policy alone does not guarantee that data collected from the individual will be used as promised. Not even P3P provides the means to enforce this within an organisation. Once data from the individual has been collected, it is up to the employees in the organisation to ensure that the data is used in a manner that is in accordance to the organisation's privacy policy. Given the size and complexity of some organisations, this is not always possible.

With this in mind, the Platform for Enterprise Privacy Practices (E-P3P) provides a system within which the promises made by an organisation, from a privacy perspective, can be formalised and enforced internally [75, 14, 18]. Essentially, E-P3P provides a means to automate an organisation's privacy guarantees by ensuring that data flow and usage practices within the organisation that involve an individual's data never conflict with the organisation's privacy policy.

Hippocratic databases

Agrawal et al [5] use the Hippocratic Oath as a basis for arguing that databases should assume responsibility for the privacy of the data maintained within them. They propose re-architecting databases to include the privacy of data as a fundamental tenet implemented through ten core principles. Essentially, the individual that owns the data and the database that collects it must specify privacy policies that detail the purposes for which data can be collected/used in addition to the recipients that can use parts of it. Agrawal et al note that Hippocratic databases go hand in hand with P3P in so far as the enforcement of an organisation's privacy policy. The XML structure of a P3P policy essentially describes what data will be collected, by whom, for what purpose and for how long. This notion of purpose and retention in P3P are directly applicable within the context of Hippocratic databases.

Rutherford et al [107, 106] take the notion of Hippocratic databases further when highlighting the privacy concerns inherent in log files. They then investigate the extent to which Hippocratic database principles can be applied in a Hippocratic log file architecture.

2.5 Conclusion

We began this chapter by highlighting a number of definitions for privacy. Having presented the notion of privacy as a concept for which there is no globally accepted definition, we eventually presented a clear definition of what is meant by privacy within the context of this research: " *the ability of the individual to protect information about himself.*"

After discussing anonymity as a form of privacy, we then discussed a number of PETs available on the Web today. Each technology presented comes with its own strengths and weaknesses. In the next chapter, we highlight these characteristics in an effort to identify which PET best addresses the search privacy problem.

Chapter 3

Search Privacy

"Ask, and it shall be given you; seek, and ye shall find; knock, and it shall be opened unto you"
Matthew 7:7

3.1 Introduction

It may be argued that many users do not employ the usage of PETs simply because they do not consider their actions online of having the potential to violate their privacy. This may be the case when one considers users that access a number of independent Web sites in a manner where no private information is ever divulged. But what of a handful of Web sites that are integral to the online experience of hundreds of millions of users each and every day? Search engines have become an important part of our experience on the Web. The content of billions of indexed Web pages coupled with billions of user queries looking for information [93] makes search engines a primary point of enquiry [12], a powerful index of what is on the Web and how relevant it may be in so far as what one is searching for.

This dependence on search engines as a source of information and a starting point on the Web today can be viewed upon as a privacy nightmare. Whilst it may not be a violation of privacy when a single day's worth of queries is stored for a single user, the implication of years of queries stored for millions of users worldwide can not be overlooked. Fortunately, there are a number of PETs that serve users in a bid to thwart some of these privacy problems. In this chapter, we will discuss what privacy means in the scope of search. Furthermore, we will focus on the relevant PETs presented in the previous chapter and discuss the strengths and weaknesses of each approach.

This chapter is structured as follows: in section 3.2 we briefly discuss the history of online search and analyse the flow of data amongst all participants concerned. Having analysed exactly what it means to conduct a search, in section 3.3 we examine the search profile stored by a search engine and discuss the privacy problem in search in more detail. Section 3.4 then looks at each of the PETs from the previous chapter and examines their applicability to the search privacy problem.

3.2 Search

The first form of search on the Internet was offered through Archie [37]. This was a database accessible via Telnet [98] that allowed users to query the content of directory listings of publicly accessible FTP [97] servers. In time, the introduction of a distributed document search and retrieval protocol (Gopher [10]) meant that users could search the contents of files and not just the names.

The growing popularity of the Web coupled with the explosive growth of its content meant that it would undergo a similar evolution. In the early days, if a user wanted to find something on the Web he/she either had to know the precise address of the document, or would have to start at a known location (for example, at the list of accessible Web servers maintained by CERN [77]) and navigate through a number of linked documents in the hope of finding something of relevance.

This laborious process of manually wading through content came to an end in 1993, when an MIT student presented the first Web search engine: Wandex [13]. This system consisted of programs on the back end called Web crawlers which navigated the Web and indexed the content found. The content was then searchable in the form of a Web front end which allowed users to query what had been searched. The Wandex search engine was shortly followed by Webcrawler in 1994 [96] and, as a result, full-text search on the Web had finally arrived.

Arasu et al [12] outline the architecture of search on the Web. This can be summarised as the interaction between three entities (illustrated in figure 3.1):

> **The user** - obviously, a user wishes to find something on the Web. Search engines facilitate this process by accepting text based queries which will result in a set of links to Web sites of relevance.
>
> **The search engine** - providing relevant links using text based queries alone is not easy. There are a number of factors that come into play when trying to determine the intent of the user (for example, when searching for the term 'free') in addition to knowing which Web sites are better than others. Search engines used to depend on well known information retrieval algorithms [46] to determine which indexed Web sites were of relevance. Unfortunately, these methods do not account for the vast amounts of information that the Web has to offer nor do they leverage off of the intricate structure of the Web. A simplified example is that of the approach which matches text that makes up a user's query directly to keywords extracted from Web pages indexed earlier. Whilst this results in a fair degree of relevance in some cases, there are a number of obvious pitfalls to this approach. Consider the query 'free', this keyword could occur the same amount of times on a simple classifieds page as it could on a page dealing with the intricacies of licensing free software. Calculating a Web site's worth or relevance with reference to a keyword or query today now employs a number of techniques that exploit the graph-like structure of the Web [7] in addition to the earlier information retrieval approaches. Of note is the Pagerank [25] algorithm, this takes into account what other pages think

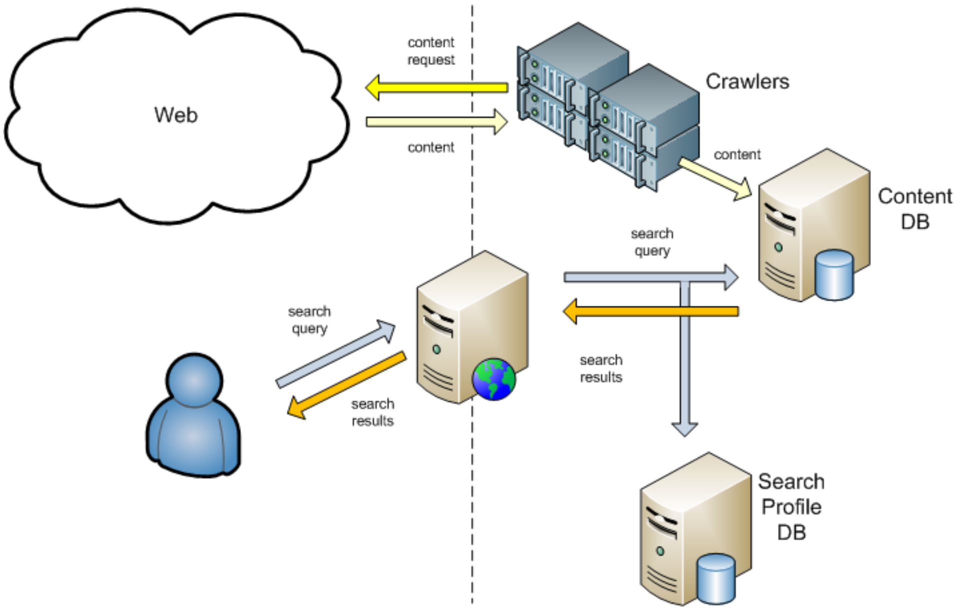

Figure 3.1: A search engine processes a user's query with the goal of delivering a set of links to relevant content on the Web that was crawled, indexed and processed earlier.

about a Web site in addition to the context used when referencing a site.

Crawlers - these programs build up search engine databases by navigating to pages on the Web and locally indexing the content found. Although they do not interact directly with the user, queries from users over time are used when deciding how often a page is revisited by a crawler. Cho et al [68] identify two types of crawlers: *periodic crawlers* which build up brand new content each time a crawl is conducted, i.e. the entire index is renewed, and *incremental crawlers* which update and refresh existing pages in the index over time.

The three entities highlighted essentially deal with the flow of two categories of information: user queries and Web site content. At its core, search on the Web is the process of determining which Web site content best matches the user query provided.

3.3 Search Profiles

It is reasonable to assume that the better a search engine is at matching the intent of the user with relevant content, then the more likely the user will be to return to the same search engine for more information in the future. The incredible efficiency and speed at which modern search engines are able to do exactly this in today's world is astounding. Finding information through the search engines has never been easier. The interesting business model employed by search engines, that of Internet advertising [43], has the search engines not only being easy, but also free. One can truly say that, for the first time, the world's information is literally at one's fingertips.

In this work, a search profile is the history of the queries issued by a user to a specific search engine. In order for a search engine to construct such a profile, it must have the ability to recognise or track users through time. There are a number methods that can be employed by a search engine to achieve this goal, Aljifri et al [8] highlight two of them:

- By recording the IP address associated with the query request. Whether fixed or dynamic, an IP address can be considered Personally Identifiable Information (PII).
- A unique cookie placed on the user's machine through the Web Browser will result in the search engine being able to recognise the user upon his/her return.

Another mechanism that may be used to track a user is that of unique characteristics in the Web request headers, for example, the User Agent string coupled with the browser's dimensions, reported location and number of sessions. Regardless of the technique used, we assume that search engines are aware of and have implemented at least one of the methods available.

The ability to track a user's queries through time means that although it is easy for a user to issue queries to a search engine, unless they are using a PET of some kind then it is just as easy for the search engine to associate the user's query with his/her profile. The nature of the Web is such that every user of a search engine can be mapped to a unique search profile consisting of the following:

- A set of search queries
- The time at which each query was issued
- Where applicable, the link that was eventually of relevance to the user, i.e., the link he/she chose to traverse to in the set of links the search

engine matched to the original query. Tracking these links is trivial, hyperlinks to relevant content are simply redirected through the search engine.

3.3.1 Offline Profiles

It can be argued that search profiles are nothing new in the sense that they are the online equivalent of a system that has been around for some time. Consider the library, users of this system are much like users of a search engine since they too are looking for information.

Online users issue a query to a search engine which results in a set of links which redirect to Web pages with more information. The onus is then upon the user to scan the content of the Web pages in order to determine if they are relevant.

Instead of a set of links on a Web page, users in a library have a number of alternatives:

- The library may classify its content using the Dewey Decimal System [38], in which case the user could go directly to the relevant section and begin going through the available sources.
- The user could ask the librarian for assistance or perhaps to make a recommendation.
- The library may have a computer system in place which has already indexed the content of its books. Finding relevant information would consist of using the system to determine which book is relevant, and then finding the book itself.

With this in mind, the search profile built up by the search engine could be compared to the history of a user in a library. In each system, there is a request for information which results in the persistence of a record. The search engine records what the user searched for (a query) and, at the very least, the library records what the user may have checked out (a book, magazine, journal et cetera). If the information stored by the library and the search engine are equivalent, then the applicable privacy regulation of a library could be applied to that of a search engine.

Of course, this is not the case. A search profile is vastly different to the checkout history in a library. When one considers the scenario of a user in a library, a search engine profile would be equivalent to a librarian following the user around and recording not only what may eventually be checked out but, more importantly, how this was found (which books were looked through) and for what purpose (which book provided the most relevance). The reason for this is simple: with the astounding growth of search engine popularity, not only are search engines used as a starting point to almost all queries for information, but search engines are able to track what the user is searching for over time in addition to what was eventually relevant.

3.3.2 A Problem

In 2006, an online search engine (AOL) made one of its own search profile databases available with the intention that it be used exclusively for research [85]. The database contained the profiles of approximately 650,000 of its users over a period of three months. Each query in the profile consisted of the following:

- The username of the user
- The time of the query
- The query itself
- Which link the user followed after the query was submitted

AOL claimed that they had *anonymised* the data by replacing the username for each profile with a random number. What followed was a privacy catastrophe: within a few days of its release the private lives of a number of users in the database were on the Web and open to scrutiny by anyone with basic SQL knowledge and a bit of curiosity. Since then, there are a number of communities and users on the Web[1] that have taken a keen interest on inferring what they can about users in this database. In some cases, users have been identified in so far as where they live, what they do and even what car they drive [19]. On the subject of releasing search profiles for research in a privacy preserving manner, a number of researchers refer to the AOL incident and propose their solutions [4, 76, 128]. Gehrke et al [60] refers to search engines as collecting a "database of intentions". In referring to the AOL debacle, they conducted a comparative study in which they analyze more sophisticated algorithms for publishing query and click histograms derived from search logs.

There is no doubt that there was a massive privacy violation with the release of the AOL database. With this in mind, we ask at what point was the privacy of these individuals violated:

[1] http://www.aolpsycho.com/, http://www.dontdelete.com/

- When the search engine kept a log of queries?
- When the search engine kept a log of queries associated with usernames and/or IP addresses?
- When the search engine kept a log of queries for more than a day, a week or a month?
- When the search engine released the queries online (after they were supposedly anonymised)?

If we consider the definition of privacy offered by Bellotti and Sellen [116] ("*a subjective notion which may be influenced by culturally determined expectations and perceptions of one's environment*") then the privacy violation could have occurred at any one of these points.

If we refer to the definition from Warren and Brandeis [119] ("*the right to be let alone*") then the violation occurred when the search engine released the queries online. It was this act that resulted in users of the database suddenly coming under the spotlight and receiving attention from the media. Had this not happened, then they would not have been bothered.

We concluded the previous chapter on the note that within the context of this work, we adopt the definition of privacy offered by Westin [120] ("*the ability of the individual to protect information about himself*"). In this definition, emphasis is placed on the ability of the individual to *control* his/her own information. If we consider a search profile as information belonging to an individual, whether personal or not, then neither of the points highlighted apply because the user in question has already lost control over the information, i.e., the search engine has it.

It can be argued that had the search profiles been anonymised more effectively then the privacy problem that followed (regardless of its definition) could have been minimised, or perhaps even avoided entirely. Despite the simple anonymisation process, one can easily look over each user's profile and draw a number of conclusions from the time a query is conducted and what other queries followed/preceded it. The problem with the anonymisation process is that each query still belongs to the original profile in its original order, i.e., each query still has context.

A lone query is not nearly as important as a query which has context. For example, a single query for a wig has a lot more meaning when it is preceded by queries for information on cancer treatment or the location of cancer clinics a few weeks earlier.

One approach to anonymising the data is that of having the queries lose their context. Instead of assigning each profile a random number, each query is assigned a random profile and only then is each profile's username replaced with a random number. This simple process of scrambling the queries is surely far more effective when thwarting the inference of private information for an individual (a profile) based on search queries alone. Of course, the usefulness of the anonymised search profiles for research purposes can be brought into question. On this topic, Götz et al [60] propose the ZEALOUS algorithm. This promises to allow search engines to publish their search logs in a manner that allows for useful research in addition to delivering a strong degree of privacy.

Cooper [28] notes that search profiles "*serve as a unique window into individuals' intentions, desires and behaviors*". In a survey of the techniques search engines could apply to their logs in order to preserve the privacy of their users, Cooper discusses the following approaches:

- Log deletion - not save the queries at all. From the perspective of a search engine taking control over the search privacy of an individual, this is highlighted as the most privacy enhancing technique available.
- Hashing queries - one-way hashes of the queries are stored instead of the queries themselves. Although this is susceptible to sophisticated attacks that exploit the frequency of queries, this technique is still considered valuable in the scenario where government agencies request query logs from search engines.
- Identifier deletion - the removal of PII the likes of an IP address.
- Hashing identifiers - much like hashing a query, this involves the application of a one-way hash function to all identifiers associated with a query.
- Scrubbing content - removing identifiers from the query, this includes Social Security numbers, credit card numbers, addresses et cetera.
- Deleting infrequent queries - proposed by Adar [4], the theory is that removing infrequent queries results in removing identifying queries. Although this approach has the potential to remove the queries of users with unusual interests, the queries are still bound to identifying data the likes of cookies or an IP address.
- Shortening sessions - privacy threats are said to have been mitigated by shortening the length of time that any identifier is associated with an individual.

Given that privacy in this thesis is directly related to the control an individual has over their own information, any approach employed by a search engine to enhance the privacy of an individual is inconsequential. For each of these scenarios, trust has to be extended to the search engine to *do the right thing*. Ultimately, the user has lost control over the search profile because the search engine has it. Since the user has no control over their search profile, the user has no search privacy.

3.4 PETs and Search Privacy

There is no question that search profiles are private. In this section, we highlight each of the PETs discussed in the previous chapter and discuss their potential for enhancing the control a user has over their search privacy. For every PET, we determine if it has the potential to enhance the control a user has over their search profile in addition to whether the search engine still poses as a threat to their privacy. We divide this discussion into the four categories of PETs listed earlier.

3.4.1 Private Communication

Defined as the ability to communicate content only to the recipient specified, PETs in this category have the potential to enhance search privacy in that only the search engine (an intended recipient of the query) will have the search profile. In the absence of this technology, there are a number of entities that may be privy to a user's search profile; for example, the user's ISP or a proxy used in the chain of servers leading to the search engine. SSL [51] is a particularly good example of a PET that fits into this category. If the Web server of a search engine supports this protocol, then users can send queries directly to the search engine with the confidence that no other entity will record the queries. Of course, the search engine still has the queries and trust is then extended to the search engine to record or transform the queries in a manner that does not violate the privacy of the individual. As a result, although PETs in this category have the potential to enhance control in terms of privacy from eavesdroppers, it does not eliminate the search engine as a primary threat to one's privacy.

This does not mean that this category should be discarded. Consider the scenario where the search engine itself is considered the eavesdropper, can there be privacy if we know who is listening and can control exactly what is sent to this entity? With reference to client-side technologies the likes of the TrackMeNot extension [66], could enough noise be generated so as to maintain the search privacy of the individual irrespective of the privacy policy of the search engine? Since users have complete control over what queries are sent to the search engine, it is reasonable to assume that sufficient noise can be generated in the search profile that will eventually be stored by the search engine. When one considers the role that obfuscation can play, this category may have the potential to enhance a user's search privacy. Whether the search engine remains a threat has yet to be discussed.

3.4.2 Anonymity

Anonymity is control over one's identity. The identity of an individual in this space may include information the likes of the IP address of the user's machine.

If a PET in this category allows a user to send queries to a search engine in a manner that does not reveal information about his or her identity, then the search engine does not know the true source of the query. Without this information, an accurate search profile cannot be constructed.

Anonymity PETs have the potential to greatly enhance the search privacy of a user. Whether or not the search engine still remains a threat though remains to be investigated. An issue that warrants further analysis is that although identifying information may not be explicitly provided, there may be the opportunity for it to be inferred. Since this is a topic of a future chapter, in this section we conclude the issue of search engines being a threat to search privacy as questionable.

3.4.3 Personal Control

This category of PET aims to ensure that any information shared by the user will be used in a manner that adheres to his/her privacy policy. For example, if the user prefers only to use search engines that do not keep any logs (store any of his queries), then PETs in this category would warn the user when this preference is in jeopardy.

Although these PETs have the potential to greatly enhance search privacy, it is not obvious how. In the case of the example above, how does one know when a search engine is not going to keep logs? Much like the previous category, we acknowledge that there is potential to enhance control over one's search privacy with this PET, but how this is done and whether or not the search engine remains a threat has yet to be investigated.

3.4.4 Organisational Safeguards

PETs in this category are aimed at the organisation, not the individual. Essentially, the organisation implements mechanisms that limit internal misuse so as to safeguard the interests of their users.

Consider the Internal Revenue Service (IRS) as an example. It may be the case that tax assessors need not be granted access to all tax returns all of the time. The IRS could implement mechanisms that only allow tax returns to be assessed by assigned individuals and only for the period for which an assessment is valid. Of course, there are exceptions (in the event of an audit, for example). In so far as issuing tax refunds to tax payers, the department responsible may only require access to their bank details and not the tax return in question. As a result, the IRS could store the bank details in one database and the tax returns in another, with separate DBAs assigned to each.

In the context of this research, the organisation responsible for implementing safeguards is the search engine. As noted in section 3.3.2, there are a number of options available to the search engine in so far as safeguarding their user's privacy. Ultimately, since this is a category of PETs that deals only with the organisation and its own internal controls, there can be no enhancement of a user's control if it has already been lost. Whether or not the user has willingly lost control over his/her search profile, the fact remains that in this category, total control over a profile lies in the hands of the organisation.

3.4.5 Summary

Table 3.1 summarises our evaluation of the PETs discussed thus far.

3.5 Conclusion

We began this chapter with an overview of search on the Web. This included a brief discussion of the history of search and an analysis of the three key entities involved: the user, the search engine and the crawlers. With a high level understanding of what search entails, we introduced search profiles. Essentially, these are logs of a search user's queries over time. In addition to storing queries, a search profile contains numerous other pieces of information; this includes the external link that was eventually followed and the IP address of the user in question.

We then introduced the privacy problem inherent with storing a search profile and referred to the database released by AOL in 2005 as an example

PET	Enhances control	SE Threat
Private Communication	Y	?
PGP	Y	Y
SSL	Y	Y
OTR	Y	Y
Steganography	?	?
Obfuscation/TrackMeNot	Y	?
Anonymity	Y	?
Anonymous proxies	Y	?
Crowds	Y	?
Onion routing (Tor)	Y	?
Personal Control	Y	?
P3P	Y	?
IE8 InPrivate blocking	N	Y
Cookie Managers	Y	Y
Organisational Safeguards	N	Y
E-P3P	N	Y
Hippocratic Databases	N	Y

Table 3.1: Each PET discussed in the previous chapter is assessed according to whether or not it has the potential to enhance search privacy and if so, whether the search engine still remains a privacy threat.

of the nature of privacy violation that a search engine is capable of. This lead us to the question of what search privacy means. Having looked at some of the definitions of privacy offered in the previous chapter, we define search privacy as *the control one has over one's search profile.* The absence of control implies the absence of privacy.

This lead to an examination of each of the PET categories highlighted earlier. We wanted to know if any of these could provide control over a user's search profile and in doing so, provide the user with search privacy. The results are unclear. The PETs that are not applicable have been easily identified but the categories that are applicable leave us asking if the search engine is still a threat. Whilst they have the potential to enhance control over a search profile, we are left unsure as to whether or not they provide enough control so as to no longer consider the search engine a threat to one's privacy.

Over the next three chapters, we examine each PET category in detail. If a category does not have the potential to provide search privacy, we investigate why this is the case and if it can be remedied.

Chapter 4

Search Privacy Through Anonymity

"Anonymity is a shield from the tyranny of the majority. It thus exemplifies the purpose behind the Bill of Rights, and of the First Amendment in particular: to protect unpopular individuals from retaliation – and their ideas from suppression – at the hand of an intolerant society."
McIntyre v. Ohio Elections Commission, 514 U.S. 334 (1995)

4.1 Introduction

In the previous chapter, we emphasised the notion that search privacy is essentially control over one's search profile. Through an anonymising network, a user could indirectly exercise control over their search profile by concealing their identity from the search engine. The premise is that if the search engine does not know the true identity associated with a search query, then it cannot construct an accurate search profile.

In this chapter, we explore the feasibility of attacking an anonymising network from a search engine's perspective (an external attack). If this is successful, PETs in the anonymity category may not necessarily be the best option for search privacy.

4.2 Motive

Popular Internet protocols the likes of TCP/IP were not designed with anonymity in mind. A TCP/IP message is likely to be routed through a collection of different servers employing logging policies that effectively expose the sender (source IP) and receiver (destination IP) of the message. As a result, entities on the Internet that are not making an effort to be anonymous are leaving trails of their activities behind regardless of what they access on the Internet or where they gain access to the Internet. Fortunately, as noted in the preceding chapter, there are a number of anonymising PETs which leverage off of current Internet protocols to provide varying degrees of anonymity [87, 27, 50, 53, 80, 90, 99, 110].

Though anonymising PETs are making it easier for legitimate users to remain anonymous, they are also making it easier for those wishing to exploit the benefits of anonymity. Examples of this exploitation include using anonymising networks to send spam and commit click fraud.

Click fraud is the intentional clicking on Pay Per Click advertising [17, 44, 121] and falls into two categories. In the first category, an attacker (possibly an unscrupulous competitor of an advertiser) generates false clicks on an advertiser's adverts in order to deplete their budget. In the second category, a fraudulent Web site owner configures a fake HTTP traffic generator [1] to generate traffic to his site (as well as click on the ads – indirectly generating revenue). Daswani and Stoppelman [34] present a case dealing with this type of fraud on a larger scale (the fraudsters used a botnet of approximately 100,000 machines).

[1] Clicking Agent, http://www.clickingagent.com

Usage of an anonymising network to commit click fraud makes it extremely difficult for the advertising company to detect whether or not the clicks being paid for are genuine. Exposing the fraudster involves exposing the users behind the anonymity network or, simply put, attacking the anonymity network.

Exposing a click fraud scheme is important in this work because it is a valid motive for a search engine. Though there may be other reasons for a search engine to attack an anonymity network, we focus on this scenario because it is feasible in the world of search as we know it today. Search engines are inextricably intertwined with advertising [70] and, as click fraud remains a threat [88], it has the potential to influence their bottom line.

Several types of attacks on anonymity have been proposed and discussed in detail within the field of anonymity networks [123, 81, 125]. Attackers have been viewed as entities that are able to monitor all traffic to and from machines on the network as well as collaborating participants of the anonymising network which pool their resources and knowledge of the network in order to expose legitimate members. If we are to be in a position so as to expose schemes the likes of click fraud, we must investigate the implications of an attack which has received little attention, i.e., an attack from outside of the anonymity network (this is the same perspective of a search engine).

The intention of this chapter is as follows: we present and discuss two principles that, when combined, may allow for a successful external attack against an anonymising network. The first principle depends on a characteristic that is common amongst anonymising networks: the unlikelihood of the sender of a message being the machine that actually delivers the message. The second principle is the ability of an attacker to recognise related requests to an end server through a shared identifier of some kind. We apply these principles in a simulated external attack against an existing anonymity model (crowds) and present the results.

This rest of this chapter is structured as follows: in section 4.3 we briefly discuss the field of anonymising technologies and provide an overview of the crowds anonymity model. Section 4.4 makes a case for attacking crowds, i.e., why it is that crowds is an eligible candidate for our attack. In section 4.5 we discuss the assumptions made for the attack against crowds. Section 4.6 then moves on to discuss the attack itself. The simulated attack is then presented, the results of which are discussed in section 4.7. This chapter is then concluded in section 4.8.

4.3 Background

Pfitzmann et al [94] define anonymity as the "*state of being not identifiable within a set of subjects*". We view the subjects in question as machines that participate on a network. An attacker plays the role of a Web server and is said to have identified a machine on the network when he can make educated guesses as to which requests on the network originated from that machine.

Pfitzmann et al [95] discuss three forms of anonymity that are possible on a network:

> Receiver anonymity - referring to the receiver of a message. The sender may be known, the message itself may be observed but the receiver of the message is anonymous (the degree of anonymity associated with the entity is discussed later in this section).
>
> Sender anonymity - referring to the entity from which the message originated. This is similar to receiver anonymity except it applies to the sender only.
>
> Unlinkability of Receiver and Sender - this form of anonymity hides the relation between the sender and the receiver of a message.

There are a number of anonymity models that employ numerous techniques to provide anonymity in one form or another. Chaum [26] employs the use of *mix* machines to delay the delivery of encrypted messages so as to hide the source of the message (typically used for email). Wright et al [122] provide an overview on a host of different mix technologies as well as on the current state of research within the field of anonymity.

Crowds [100] offers various degrees of anonymity through the notion of blending into a crowd. The degree of anonymity, with respect to the sender of a message, is defined as an informal continuum ranging from *absolute privacy* to *probable innocence* and eventually *provably exposed*. Refer to [39] for more detail on quantifying anonymity.

Each member in a crowd (a jondo) receives and forwards messages on behalf of any other member in the crowd. When a jondo decides to send its first message via the crowd, a random path must be configured to serve as the route taken for all messages sent by that jondo from that point onwards. The route is established as follows:

1. The sender picks a random jondo and forwards the message to him.

2. This jondo then flips a coin which determines whether to forward the message to yet another jondo (in which case this step is repeated) or to send the message to the end server.

Upon receiving a response from the end server, the message is routed back along the path to the initial sender of the message.

The authors of crowds define three types of attackers:

- A local eavesdropper is defined as someone that can monitor all communication from a single computer (jondo). Though no sender anonymity is offered from this type of attack (since the attacker can see all messages initiated from the computer), Rieter and Ruben describe how receiver anonymity (the end server) tends towards *beyond suspicion* as the size of the crowd tends towards infinity.

- Collaborating crowd members are seen as jondos that pool their information regarding the crowd together in an effort to expose crowd members. An attack is described where collaborators on a path of jondos used to send a message try to determine if their immediate predecessor on the path is the sender. Reiter and Rubin conclude that *probable innocence* is guaranteed for sender anonymity as long as $n >= 3(c+1)$ where n is the size of the crowd and c the number of collaborating jondos. Wright et al [124, 123] formally define this type of attack (the *predecessor attack*) and examine it within a broader scope.

- The end server: Rieter and Rubin make a strong case for sender anonymity against an attack from the end server. They suggest that the anonymity of the initial sender of the message is *beyond suspicion*.

Since this chapter concerns itself with an attack launched from the end server (an external attack), we discuss details of this attack and assumptions made for the attack in sections 4.5 and 4.6.

4.4 The Case for an External Attack

In this section we discuss the elements that contribute to making a case for attacking crowds. The two principles that must apply to make crowds an eligible candidate for an attack are as follows:

1. The unlikelihood of the sender being the machine that ultimately delivers the message.

2. Recognition of related requests through the usage of a shared identifier.

In satisfying these two principles, we move on to discuss the attack itself.

4.4.1 Principle 1 - an unlikely sender

Reiter and Rubin [100] state, that from an end server's perspective, each member of a crowd is equally likely to have been the sender of a particular request. If I denotes the event where a member of the crowd initiates a message and S the event where a member of the crowd sends the message then $P(I \cap S|k) = 1/n$ where k is the path length (dependent on the probability of forwarding a message in the crowd) and n the number of members in the crowd.

Note that the value of k actually has no impact on the probability of the sender delivering his own message. The reason is that at step $k - 1$ the currently chosen member of a crowd must decide where to forward the message to and since he is as eligible a member as anyone else in the crowd, including the sender, it is reasonable to say that the probability of the sender being chosen is $1/n$. Since the decision taken at step k - 1 is independent of the size of k and will occur for all values of k greater than 1, we say $P(I \cap S|k) = P(I \cap S)$.

This suggests that as n tends towards a significantly large number it is unlikely that a participant in a crowd will issue requests to an end server when it is also the sender of the request. The first principle of our attack has therefore been satisfied.

4.4.2 Principle 2 - recognising related requests

The first principle allows an attacker to start making assumptions about the machines that are part of the anonymising network (the crowd in our case). Being in a position to make these kinds of assumptions is not entirely sufficient when attempting to compromise the anonymity of the sender though; a mechanism to recognise requests must exist, i.e., a way in which to relate requests to one another (and ultimately the sender of the requests).

The mechanism adopted in this chapter is generally accepted by the Web community and is in widespread use on the Web as we know it today: that of Web servers issuing cookies to Web browsers [2].

Cookies allow Web servers to maintain state and effectively recognise repeat visits from users to a given URL. When a new user requests a page from a Web site, the response headers may include a unique identifier that the user's browser would then store in its cookies file on the hard drive. Further requests to the Web site will include this identifier. The usage of cookies will ensure that all related requests share a unique identifier. Although the

[2] Using Cookies with CGI, W3C Journal Volume 1 (available online at http://www.w3j.com/4/s3.shishir.html)

authors of crowds suggest disabling cookies on the jondo agent, support is available for it.

By supporting cookies in a crowds environment we have satisfied our second principle for an external attack against an anonymising network.

4.5 Assumptions

In our attack against crowds we make four assumptions:

1. All participants in the crowd have cookies enabled.

2. All participants forwarding requests in the crowd will not strip out cookies.

3. There is a large enough group of colluding servers to ensure that a large number of requests from the crowd are forwarded to at least one of these servers. The large group of colluding servers allows us to define a cookie farm: a depot from which colluding servers lookup and assign cookies to participants in the crowd.

4. In issuing cookies from the cookie farm to members of the crowd, the colluding servers are able to learn which machines (jondos) are part of the crowd. This assumption is not entirely unreasonable. If a machine that has already been handed a cookie starts making requests to the colluding servers using a different cookie (or without any cookie at all), the colluding servers can mark the machine as a potential member of a crowd (due to it acting as a proxy for other machines).

Since it is unlikely for a participant p_1 to represent himself when making a request to an end server, we can say that it is equally unlikely that a cookie issued to participant p_1 actually belongs to p_1.

When the colluding servers receive a request from a participant in a crowd where no cookie has been issued, they are now in the position to make the following assumption: in issuing cookie C to participant P it is highly unlikely that C actually belongs to P. The postulate drawn is therefore as follows:

For a significantly large crowd, P_i is not the owner of C_k

where i denotes any member of the crowd that has made a request and k denotes the cookies that were issued to P_i.

4.6 The Attack

This section describes the attack through the aid of an example. The example presented uses a small crowd consisting of three participants (p_1, p_2, p_3). Although the first principle presented in this chapter relies on a significantly large crowd, we use a small crowd so as make it easier to appreciate the gist of the attack.

When participant p_1 initiates a request for the first time in the crowd he will typically select a random member in the crowd (let this be p_2) to whom the message will be forwarded. This member will then decide whether or not to send the request to the server in question or forward it to another random member of the crowd.

The act of forwarding the message to a random member versus sending it to the server results in the configuration of a random path used to route messages through for the sender. Upon establishment of the path, it will serve the sender until the central authority of the crowd (the *blender* [100]) issues a command to all participants to reset their paths, for example, in the event of a participant joining the crowd. Table 4.1 depicts a mapping of senders (S) to their respective participants that will ultimately be responsible for delivering the message (S').

S	S'
p_1	p_2
p_2	p_3
p_3	p_1

Table 4.1: A map of senders and participants responsible for delivering the message.

The assumptions made in section 4.4 allow the colluding servers to determine if the sender represented by S' has been issued a cookie from the server. Since the cookie farm is a central repository of cookies and all colluding servers are using the cookie farm, the colluding servers are able to begin mapping out which cookies have been issued to which participants in the crowd. Under the postulate that it is highly unlikely for a sender to represent himself, the colluding servers can say with a fair degree of certainty, that since cookie c_1 has been assigned to participant p_2 it is unlikely that cookie c_1 actually belongs to p_2, i.e., the colluding servers can start building sets of cookies that have been assigned to participants and moreover, sets of cookies that are likely to belong to participants. Let A_i denote the set of cookies handed to p_i. In our example the servers know the following:

$A_1 = \{c_3\}$
$A_2 = \{c_1\}$
$A_3 = \{c_2\}$

Let B_i denote the set of cookies that p_i could simply not be (for example, the case where new cookies are generated and handed out to new members joining the crowd). If we define C_i as the set of cookies that p_i is likely to be then, using our small sample and concentrating on p_1, we can say

$B_1 = \emptyset$
$C_1 = \bar{A}_1 \bigcap \bar{B}_1 = \{c_1, c_2\} \bigcap \{c_1, c_2, c_3\} = \{c_1, c_2\}$

With the crowd we have described thus far, the colluding servers are in a position to make the following assumptions about each member:

p_1 does not have the cookie c_3 it probably has cookie c_1 or c_2.

p_2 does not have the cookie c_1 it probably has cookie c_2 or c_3.

p_3 does not have the cookie c_2 it probably has cookie c_1 or c_3.

Now consider the scenario where another participant joins the crowd. Because the paths used by each of the senders has to be "forgotten", it is not likely that the same participant will be mapped to the same sender when making future requests to the colluding servers. If the cookie issued to each participant S has not been deleted upon the crowd resetting then let table 4.2 depict the routes that have been configured as a result of another jondo joining the crowd.

S	S'
p_1	p_3
p_2	p_1
p_3	p_4
p_4	p_3

Table 4.2: A map of senders and participants upon a new jondo joining the crowd.

Since new participants are now representing the same cookies, the servers are in a better position to make assumptions as to which cookies belong to which machine(s) from the crowd. With what the servers know about the cookies assigned to the crowd, and with the postulate that a cookie assigned to a participant probably does not belong to the participant, we know:

$A_1 = \{c_3, c_2\}$
$A_2 = \{c_1\}$
$A_3 = \{c_2, c_1, c_4\}$
$A_4 = \{c_3\}$

$B_1 = \{c_4\}$
$B_2 = \{c_4\}$
$B_3 = \{c_4\}$
$B_4 = \{c_1, c_2, c_3\}$

$C_1 = \bar{A}_1 \bigcap \bar{B}_1 = \{c_1\}$
$C_2 = \bar{A}_2 \bigcap \bar{B}_2 = \{c_2, c_3\}$
$C_3 = \bar{A}_3 \bigcap \bar{B}_3 = \{c_3\}$
$C_4 = \bar{A}_4 \bigcap \bar{B}_4 = \{c_4\}$

Note that B_1, B_2 and $B_3 = \{c_4\}$. We know each of these participants could not possibly have c_4 since this cookie did not exist when they were part of the crowd. Evidently, this attack is much harsher on new members to the crowd. In the case of p_4, the colluding servers know that it cannot be holding cookies c_1, c_2 or c_3 since the machine was not part of the crowd when those cookies were handed out. Therefore, they can with a fair degree of certainty assume that $B_4 = \{c_1, c_2, c_3\}$ and as a result, p_4 is holding the c_4 cookie.

4.7 A Simulation

To test the effectiveness of this attack a simulator was written. The simulator implements the behaviour of a crowd on the one hand as well as the colluding servers on the other. Each test using the simulator is carried out as follows:

1. Instantiate a crowd of size n.

2. Initiate a single request from each member of the crowd to the colluding servers (assuming all members of the crowd send approximately the same amount of traffic [48]). In making an initial request, the random path for each member is configured.

3. If a colluding server receives a request from a member of the crowd that does not have a cookie then a cookie from the cookie farm is issued

to it. The colluding servers keep track of which cookies were handed out to which members of the crowd.

4. The colluding servers try to make assumptions about the crowd, i.e., which cookies belong to which members.

5. A new member is added to the crowd.

6. If the crowd is below a predefined limit the test proceeds to step 2, otherwise the test is complete.

As discussed in section 4.4, the number of jondos that participate in the routing of a message (the path length) has no impact on the probability of delivering one's own message. Since the probability of forwarding a message from one jondo to another directly influences the path length, this value was constant for each of the tests and set to 0.5.

Figure 4.1 depicts the results of the tests conducted. The percentage axis indicates the ratio of assumptions made by the colluding servers that were correct. The α axis depicts the degree of certainty required by the colluding servers before making an assumption.

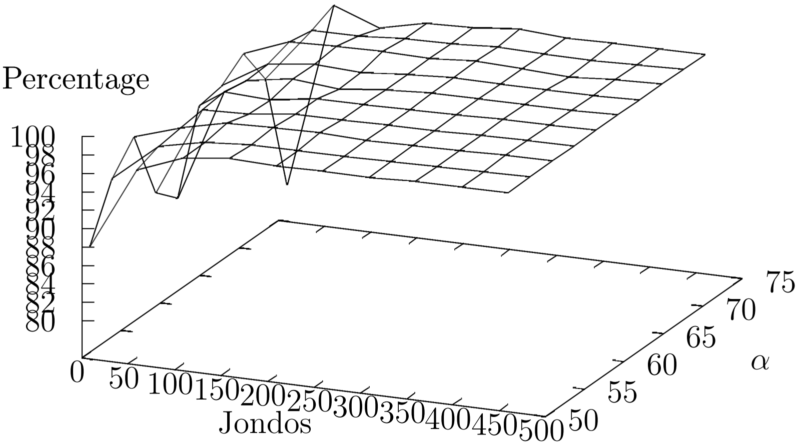

Figure 4.1: Correct assumptions made by the colluding servers

In the case where 200 cookies have been issued to the crowd, with $\alpha = 70\%$ the colluding servers will only try to make an assumption for any member of the crowd (predict which cookies match to which machines) when the number of cookies in the prediction is less than 70% of the cookies handed out. In this case, less than 60 cookies.

Figure 4.2 depicts (as a percentage) the number of crowd members that were exposed for each attack. This suggests that most of the members being exposed in the crowd are new members to the crowd (in doubling the crowd, 50% of the members in the crowd were exposed). Closer analysis of the assumptions made by the colluding servers has revealed that this is indeed the case. With the crowd size only doubling in size there are very few original crowd members that were exposed by the colluding servers.

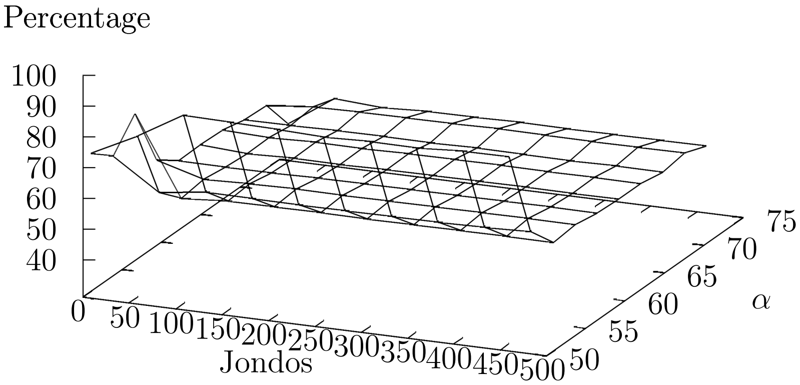

Figure 4.2: Percentage of crowd members that were exposed through an attack.

With this in mind, we conducted another test where we grew the crowd past the double original size limit. Figure 4.3 depicts the results of this test and clearly shows that the number of original members exposed by the colluding servers tend to be far greater as the crowd keeps growing.

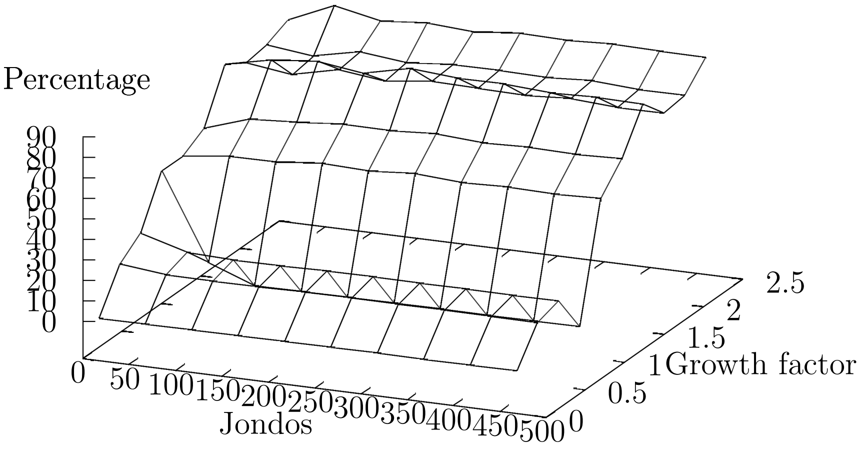

Figure 4.3: Percentage of the original crowd that has been exposed through an attack. A growth factor of 1.0 indicates that the crowd grew by 100% of its original size.

4.8 Conclusion

We began this chapter by discussing the motive for externally attacking an anonymising network. We then presented two principles which, when combined, may result in successfully exposing members of an anonymous network by means of an attack which has received little attention.

The first principle of the attack relies on the sender of a message not representing himself when delivering the message to an end server (this is common in anonymity networks). The second principle depends on a mechanism for recognising related requests; the mechanism used in this chapter was that of cookies. In issuing cookies to members of the crowd under the postulate that a cookie issued to a participant probably does not belong to that participant, the external attacker is able to start creating sets of cookies to which a candidate is likely to belong.

The attack discussed in this chapter was tested by means of a simulation. The results showed that with sufficient growth of the crowd, a significant number of original crowd members may be exposed. The impact of these results may serve so as to shed some light on the often overlooked external attack. This is especially true for crowds. When considering an end server (external attacker) the authors of crowds rate the anonymity of a participant sending a message as *beyond suspicion.* The results of the attack proposed in this chapter suggest that the anonymity of the participant tends more towards *possible innocence*, i.e., "there is a nontrivial probability that the real sender is someone else".

In this chapter, we have shown that under certain conditions the usage of an anonymising network does not guarantee anonymity from an external party (the likes of a search engine). Since search engines have a motive for attacking an anonymity network, the result of which may be successful, it is reasonable to assume that anonymity PETs are not the ideal environment for search privacy.

Chapter 5

Search Privacy Through Personal Control

"I am the master of my fate, I am the captain of my soul"
William Ernest Henley

5.1 Introduction

In chapter 3, we looked at each PET category and asked two questions:

- Does it have the ability to enhance a user's search privacy?
- Is the search engine still a threat to the user's search privacy?

Of the categories analysed, some showed potential but we were unable to provide conclusive answers. In the previous chapter, we investigated the anonymity category in an effort to determine if it can enhance a user's search privacy as well as eliminate the threat of the search engine. Having control over one's identity does contribute to enhancing search privacy, if the search engine does not know who is submitting the query then an accurate search profile cannot be constructed. However, we then showed that the search engine may have a motive for determining who is submitting the query (in the case of thwarting click fraud) in which case anonymity may not be guaranteed. As a result, while PETs in the anonymity category do have the potential to enhance a user's search privacy, they do not eliminate the search engine as a threat to the user's search privacy.

In this chapter, we turn to the category of personal control. Highlighted as "the use of technology to ensure that an individual's personal information is only used in a manner commensurate with the individual's privacy policy", we will determine if PETs in this category can adequately address the issue of search privacy. To evaluate this PET category, we consider a technology that embodies its definition: the Platform for Privacy Preferences (P3P) [32, 31].

P3P allows Web sites to declare their intentions as far as privacy-related matters are concerned. Users of the Web, having explicitly configured their own privacy preferences, are then able to browse through Web sites that have P3P policies with the confidence that their privacy will not be violated.

Although P3P has made an important contribution to the field of privacy, there are two problems:

- Trust: how does one trust what a Web site has published in its P3P policy? Whilst a legitimate Web site has every reason to publish a P3P policy stating its intentions, there is obviously no incentive for an illegitimate Web site to publish its suspicious privacy practices in a P3P policy. Therefore, if an illegitimate Web site publishes a P3P policy at all, why should it not also publish a policy that states its intention of respecting every user's privacy, despite the fact that it will do nothing of the sort?

- Proxies: although the way in which users and their associated P3P agents interact with a P3P compliant Web site has received much attention, the surreptitious role played by the Web proxy server and its impact on P3P has been neglected. We must consider the scenario involving a Web proxy that is situated between a Web user and a P3P compliant Web site.

Without addressing the issue of trust and proxies in P3P, we can go no further in its evaluation as a technology in the personal control category. This chapter is therefore structured as follows: in the next section we briefly look at the design of P3P before dealing with the issue of trust in section 5.3. Section 5.4 then deals with proxies and their place in P3P. This chapter is concluded in section 5.5.

5.2 P3P

P3P has been the focus of much research and criticism [29, 31, 32, 65, 114] and at its most basic level allows Web users (with their associated P3P agents) to automate the protection of their privacy. Web sites publish P3P policies clearly describing their intentions so that Web users can compare these policies to their own set of privacy preferences. Provided that the P3P policy published by the Web site is acceptable, the user may continue to make use of services offered on the Web site. The P3P policy of a Web site typically states what information it may require from a user during a session as well as what it intends to do with the information. The gist of a user's privacy preferences will describe what type of personal information the user is willing to provide, how long it may be stored, for what purpose this information may be used and to whom it may be given.

P3P policies are published in a standardised XML format. Although it is quite possible to peruse them manually, the process of going through a policy is typically automated through the use of an agent that is familiar with the user's preferences. An example of a P3P policy implemented by a Web site is as follows:

```
<POLICIES xmlns="http://www.w3.org/2002/01/P3Pv1">
  <POLICY discuri="http://example.com/privacy.html"
          name="An example of a policy">
    <ENTITY>
      <DATA-GROUP>
        <DATA ref="business.name">
          Example Web Site
        </DATA>
      </DATA-GROUP>
    </ENTITY>
    <ACCESS>
      <nonident/>
    </ACCESS>
    <STATEMENT>
      <CONSEQUENCE>
        We keep standard access logs but everything
        is anonymized.
      </CONSEQUENCE>
      <NON-IDENTIFIABLE/>
    </STATEMENT>
  </POLICY>
</POLICIES>
```

Note the `<NON-IDENTIFIABLE/>` tag, this indicates that information stored by the Web site cannot be traced back to the individual.

P3P policies can be located via Policy Reference Files. These files are responsible for defining which P3P policies apply to certain URIs. Cranor *et al* [31] specify four methods that may be used to find the Policy Reference File that will then be used to look up the appropriate URI:

- It may be located in a *well known* location, for example, http://example.com/w3c/p3p.xml
- A document may indicate a policy reference file through an HTML *link tag.*
- A document may indicate a policy reference file through an XHTML *link tag.*
- The reference file may be indicated through an HTTP header.

5.3 Trust and P3P

Fortunately, the problem of trust is not new to the Web. In the past, environments such as online trading systems needed a mechanism to establish trust between a buyer and a seller that were relatively unknown to one another. This was achieved with the introduction of a Reputation System. Users of the system were granted reputations that could be built up or broken down depending on how they behaved.

In this section, we discuss the problem of associating trust with Web sites within the context of P3P, that is, we identify the need for a Reputation-based System in P3P. We analyse Reputation Systems in detail and discuss the potential problems of extending P3P with a Reputation-based System. As a result we offer an alternative to conventional Reputation Systems in the form of a trusted third party.

5.3.1 Reputation Systems

The trust between a buyer and a seller in a face-to-face sale can be established through several means. Before paying the seller, the buyer must be able to determine whether the goods for sale are, at the very least, genuine. If the goods are acceptable, the buyer must next determine if the seller is trustworthy, i.e., will the goods be delivered after payment has been made? Of course, the transaction is sure to run with fewer problems if both parties have credible reputations and trust each other.

Since the normal elements associated with a face-to-face sale (holding or seeing the goods for example) are not present in an online environment, other mechanisms must be relied upon to establish some degree of trust between the two parties involved before any transaction can occur. This degree of trust depends largely on the reputation of the entities involved. Josang et al [73] point out that although there is a close link between reputation and trustworthiness, it is evident that there is a clear and important difference. In this work, we adopt the definition Josang et al have chosen from the Concise Oxford Dictionary: *reputation is what is generally said or believed about a person or thing's character or standing.*

What is said about an entity, either good or bad, is usually taken from previous dealings and interactions with that entity. Reputation Systems are responsible for gathering these opinions and forming quantifiable reputations for users of a particular system. They serve as a mechanism to establish a degree of trust between participants in the system and do this by collecting, distributing and aggregating feedback regarding participants' past behaviour [102].

As an example, eBay is briefly examined as one of many online trading business-to-consumer and consumer-to-consumer environments. eBay's Reputation System is called the Feedback Forum [102]. Opinions about sellers and buyers come in the form of feedback which is given voluntarily by other buyers and sellers after a sale. Feedback can be either good (+1), bad (-1) or neutral (0). Neutral feedback is also considered part of a feedback system. Feedback points are computed by taking the number of unique users who left good feedback and subtracting the number of unique users who left negative feedback [101]. One seller is said to have a better reputation than another seller if the ratio of positive to negative feedback is better.

Consider sellers Alice and Bob. Alice conducted 500 transactions with only one negative feedback. Because of the ratio of negative to positive feedback Alice has a good reputation. Bob also conducted 500 transactions but received 50 negative feedbacks. Although 90% of Bob's feedback is positive and he is probably a trustworthy seller, he does not appear to be as trustworthy as Alice. In other words, Alice has a better reputation than Bob.

In online trading systems, a buyer will most likely have no idea who the seller is or what the condition of the goods being sold are (other than the condition the seller claims them to be in). With a Reputation System in place, the potential buyer has an accurate idea of how trustworthy the seller is based on their reputation.

Reputation Systems can be divided into two types of architectures: Centralised Reputation Systems and Distributed Reputation Systems [73]. Centralised Reputation Systems have a central authority responsible for the construction of a reputation for a user. A Distributed Reputation System is one in which agents participating in the system are responsible for constructing a user's reputation (e.g. Peer-to-Peer Reputation System) [63, 3, 45].

5.3.2 The Need for Trust in P3P

In section 5.3.1 the critical role performed by Reputation Systems within the context of online trading systems was highlighted. With a Reputation System in place, buyers in the system can easily associate varying degrees of trust with sellers who are generally unknown to them. Without a Reputation System, buyers have no immediate mechanism at their disposal to assist them in deciding who may be trustworthy and who may not. P3P is no exception to the problem of determining who to trust. A potential user of a Web site with a P3P policy may have had no experience with the Web site in the past and as a result has very little from which to judge whether it can be trusted and the Web site's adherence to their published P3P policy.

The intention of P3P is to allow potential users of a Web site to make informed decisions regarding their privacy needs. But how does a user decide whether or not to trust the Web site in question? This Web site may be actively participating with a syndicate that collects personal details (such as full names and email addresses) for the purposes of distributing spam. Despite its unscrupulous approach towards the privacy of its users, the Web site may have a sterling P3P policy that indicates nothing of the sort.

Whilst P3P makes it easier for users to determine if the privacy policy presented by the Web site they are visiting suits their privacy needs, it does not guarantee that the policy presented by the Web site is legitimate. To be precise, there is no guarantee that the actual privacy practices of the Web site are indeed as claimed in its P3P policy.

With the introduction of a Reputation System, trust in a Web site would depend not only on whether or not its P3P policy is acceptable, but also on the reputation associated with the Web site in question. Obviously, a Web site with a bad reputation is less likely to uphold its privacy promises than a Web site with a good reputation.

Implementing a Reputation System within the context of P3P implies several challenges. To better appreciate these challenges, we first examine and discuss the prerequisites of a Reputation System.

5.3.3 Prerequisites of a Reputation System

Various Reputation Systems are available on the Internet: eBay[1], Amazon[2], Yahoo[3] and Bizrate[4]. Common usage includes rating recommendations, articles, buyers and sellers. They are also used by businesses to rate customers and vice versa. Essentially, a Reputation System's main function is the collection and distribution of feedback.

According to Zacharia and Maes [126], one method of building a Reputation System involves the creation of a central agency that records the recent activities of users in the environment. A centralised system monitors users' activities and provides a summary thereof to other users. The centralised system also accepts feedback or comments in order to compute reputation points for each user.

To evaluate the effectiveness of a Reputation System we must examine its properties. Resnick et al [102] list the following three requirements that a Reputation System must meet in order to operate effectively:

- Long-lived entities that inspire an expectation of future interactions
- Capture and distribution of feedback regarding current and past interactions (this information must be visible in the future)
- Use of feedback to guide trust decisions

[1] eBay, http://ebay.com

[2] Amazon Auction, http://auctions.amazon.com

[3] Yahoo! Auction, http://auction.yahoo.com

[4] Bizrate Online Business Ratings, http://www.bizrate.com/ratings_guide/guide.html

In online communities, a low reputation may be the result of a number of reasons (including fraud). Users who receive low reputation points often leave the system and obtain another identity for the purpose of starting afresh with a brand new reputation. This obviously violates the properties of a Reputation System which is built on the main idea that once a user becomes part of the community he or she starts accumulating reputation points. These points are acquired through interactions with other online community users. The users provide feedback to a central reputation system so that other users may have access to such feedback or evaluations to help guide their decisions.

The alternative to a centralised Reputation System is a distributed Reputation system. In this system, users do not submit their feedback to a central authority. Instead, they record their opinion of each experience with other users and provide this information at the request of a "relying party" [73]. When users need reputation information they must find the distributed community of users who have already had direct experience with the specified user. Each user is responsible for computing its reputation points based on interactions with users and private interaction with a target user. Note again that feedback is not submitted to a central authority.

A theme that has received much attention throughout this section is that of a feedback intensive environment. If there is no feedback on a user, reputation points cannot be computed for the user. In other words, constant interactions are required to enable an authority to compute reputation points.

The availability of reputation points also defines a good Reputation System. Users must be able to know that a target user is potentially trustworthy and as a result make a trusted, informed decision. Resnick et al [102] conclude that systems which rely on the participation of large numbers of individuals accumulate trust simply by operating effectively over time. The effectiveness of a Reputation System relies mainly on the honest participation of its community members.

5.3.4 A Reputation System in P3P

We now consider conventional Reputation Systems in a distributed environment like the Web, their properties and why they will fail in P3P. We subsequently propose an alternative to the conventional system as well as a means to visualise (and formalise) reputation.

Conventional Reputation Systems and P3P

Conventional Reputation Systems are built on a centralised platform for the convenience of users who submit and request feedback or reputation points. The Reputation System used in eBay is well studied [73, 101, 102, 126] and several Reputation Systems have been built upon this concept (Yahoo! Auction and Amazon Auction for example). In this section we attempt to model a Reputation System for P3P in an effort to further legitimise the P3P policy presented on a Web site.

A Reputation System within the context of P3P would be responsible for accepting feedback from registered or frequent users of a Web site with respect to the effectiveness of the Web site's privacy policy. For the sake of simplicity feedback provided by users of the Reputation System can be either positive or negative.

Each Web site is obliged to display its reputation points alongside its privacy policy (or provide a link to the central system which contains the score). Thus far, the simple system presented satisfies the three properties of a Reputation System as listed in section 5.3.3.

Unfortunately, the system suffers from two problems. The first problem is the centralised nature of the Reputation System. Since the nature of the Web is distributed, there are a number of difficulties in having a central Reputation System where users can provide their opinion on every Web site's P3P policy.

Even if feedback were only for e-Commerce Web sites, it would still be tremendously cumbersome (the number of e-Commerce sites grows daily). The alternative may be a distributed Reputation System but again the vastness of the Web would make it close to impossible to locate a user to either log or obtain reputation points. The time involved in looking for a user who has a reputation score for a Web site would take up system resources and therefore make it too costly [73].

The second problem with the system presented involves the capturing of feedback. The difference between a trading system and P3P is that there is no tangible product being delivered upon accepting the P3P policy of a Web site. How will a user determine that his/her privacy has been violated by a site that is not adhering to its own P3P policy? We could specify a time period for a user to provide feedback after visiting a Web site, but no definite time period can be specified because the Web site under review may decide to violate its privacy policy (sell a user's details, for example) long after feedback has been provided. Yet another problem is identifying a legitimate user if a definite time period is defined. Registered users can be identified because they have a profile and probably a history. However, unregistered users may visit a Web site once to complete a transaction and are never seen again.

Unorthodox Reputation Systems and P3P

We have discussed some of the problems concerning the usage of conventional Reputation Systems and their application to P3P. As a solution to these issues, we propose a Reputation System that is based on the dependence of third parties. As mentioned earlier, the distributed nature of the Web causes a centralised Reputation System to be infeasible. A Reputation System built and maintained by the same entity that a user wishes to query would not be feasible either. Extending trust to third parties is a mechanism that is already in use today (through services such as BBBOnLine[5]). We argue that trust in a third party will grow as the number of Web sites supported by the third party increases.

It was pointed out earlier that a Reputation System without feedback (opinions) from entities within the system is essentially useless. In the previous section we discussed the difficulty of providing feedback within the context of P3P. However, if a Reputation System is going to be successful in P3P, it has to cater for feedback of some kind.

We therefore offer an alternative to feedback. We believe that a Reputation System based on the two factors presented in this section will suffice to replace the type of feedback one is generally familiar with in a conventional Reputation System. These two factors are maintained by the third party and are as follows:

- The length of time that the Web site has been registered with the trusted third party.
- The number of confirmed incidents reported against a Web site, i.e., incidents where the Web site has been proven to have violated its own P3P policy.

We believe that a reputation with a 5-year history is more credible than a reputation with a 5-month history (not counting exceptional cases). Building up a reputation as a function of time essentially removes the dependence of the Reputation System on direct feedback alone.

Feedback in the form of confirmed reports cannot be constrained by short time limits, for example, it may be the case that months pass before it is found that a Web site has violated the privacy of a user (perhaps in the

[5] A Better Business Bureau Program - http://www.bbbonline.org

form of unsolicited email). In this example, the user would then report this incident to the trusted third party. After an investigation, the report may be confirmed, in which case the reputation of the Web site in question will be affected negatively. A simple formal solution is discussed in 5.3.4.

Preparing and Presenting Reputation

In our proposed Reputation System, the length of time a Web site has been registered with a third party is an important factor in determining its reputation. Simply presenting a user with graphs depicting time lines, the number of incidents and the confirmed number of incidents (although factual, up to date and accurate) may not necessarily be that helpful. We therefore propose an alternative method to calculate and present or visualise a reputation.

The mechanism used for presenting a reputation can be compared to the cross section of a tree trunk. Each ring is indicative of a period of time. The longer the period of time that a site has been registered, the more rings it will have. For our purposes, a ring can be thought of as a one-month period. Note that rings grow outwards (like in a tree) and that the thickness of the rings (the black outline) decreases as the number of rings increases. We refer to this type of reputation mechanism as the Reputation Ring. Figure 5.1 depicts the Reputation Ring of a user in its early stages.

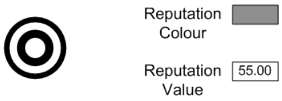

Figure 5.1: Reputation Ring of a new site.

From a presentation point of view, reputation is effectively a measurement on a colour scale ranging from black (completely untrustworthy) to white (trustworthy). The value used to represent the reputation of an individual is a ratio of the number of white pixels to black pixels in the Reputation Ring.

A site's reputation in this system differs from conventional Reputation Systems since new entities in the system do not start off with sterling reputations. Exceptional or flawless reputations have to be earned over time (as depicted in figure 5.2).

A confirmed privacy violation report against a site would darken the applicable ring by a configurable shade of gray. In figure 5.2 the third ring (third month in this case) has been darkened by one shade because of a confirmed report against the Web site. The effects of confirmed reports fade

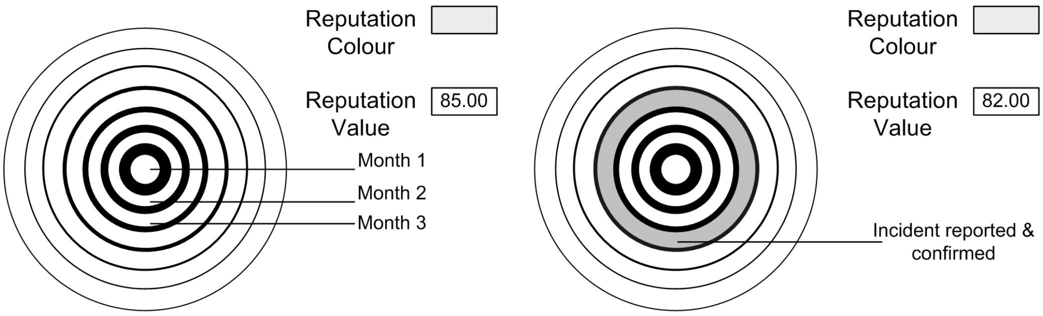

Figure 5.2: A reputation that has improved over time and a similar reputation that has had a confirmed incident reported against it.

as time progresses (with the introduction of more rings there will be more white pixels and so a better ratio).

Formalising a Reputation

A reputation is said to be the ratio of positive feedback to negative feedback. Since feedback in our system is measured in the form of time as well as in the number of confirmed reports against an entity we say reputation is comprised of variables a and b.

Variable a is defined as positive feedback while b is defined as negative feedback. The reputation for a single Web site can be calculated in one of two ways:

1. For a particular period in time. This is similar to calculating the ratio of white to black pixels for a particular portion of the Reputation Ring.

2. The overall reputation of the Web site. This is ultimately what a user would want to see.

For the sake of simplicity we calculate reputation in monthly intervals. And since reputation is the ratio of positive feedback to total feedback, the reputation for month t is defined as follows:

$$R(t) = \frac{a}{a+b} = \frac{a+b-b}{a+b} = 1 - \frac{b}{a+b} \tag{5.1}$$

where a, b are positive real numbers

Since a is a measure of positive feedback and this is proportional to time we denote a at t as $a(t) = t$ where t is a unit of time (equal to the number of months a user has been registered).

We define $b = c + v * r$ where c is a constant that all reputations start off with and can be improved over time; r is the number of confirmed reports against a user and v the desired impact of a confirmed report on a user's negative feedback (this can be compared to introducing a shade of gray to the Reputation Ring). As a result

$$R(t) = 1 - \frac{c + v * r}{t + c + v * r} \tag{5.2}$$

The overall reputation of a Web site is the average of the sum of the reputations over the time it has been registered i.e.

$$\overline{R} = \frac{\sum_{i=1}^{t} R(i)}{t} \tag{5.3}$$

The simple calculation of reputation offered in this section allows for a site's reputation to improve over time. Although reported incidents may have an impact on a reputation because of the way in which we have defined negative feedback (as a weighted constant) the effect of incidents will fade as time passes if the user does not have any more negative feedback.

Future Work

What must be investigated further is the weight of v. How much impact should a confirmed report have on a user's reputation? We gradually increase v from 0.1 to 1.0. Note that although v has more of an impact on the monthly reputation (Figure 5.3) there is not as much of an impact on the overall (Figure 5.4) reputation.

What is of great importance is whether or not v plays any role in the following scenario: user U_1 has been a part of the system for some time. At time T_1 another user U_2 joins the system. A confirmed incident is reported against U_1 at the same time. Once some time has passed, should U_1 and U_2 have the same reputation? Could v play a part in determining the outcome or could it be the case that the nature of a will have more of an impact than v?

5.3.5 Reputations Systems Bring Trust to P3P

In this section, we identified the problem of trust within the context of P3P. Although P3P allows Web sites to describe their privacy practices in a structured manner (in the form of a policy) there is no mechanism to guarantee

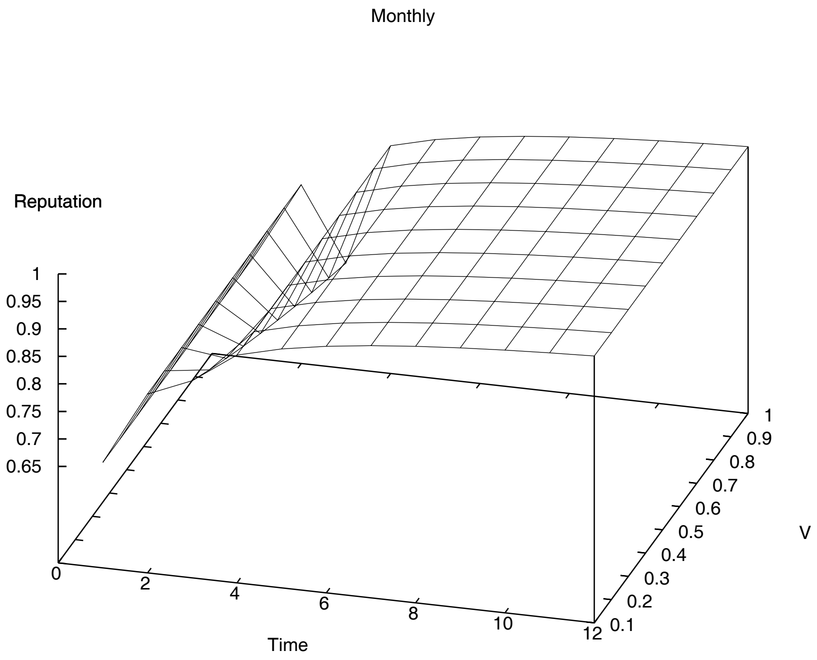

Figure 5.3: The effect of v ranging from 0.1 to 1.0 using a 12 month reputation with a single confirmed report against it.

users that a Web site will adhere to the P3P policy specified. It was suggested that a solution to this problem may lie in the form of a Reputation-based System. Although a Reputation System does not provide a user with any guarantees, it does give a user enough information so as to better decide how much trust to associate with a Web site.

Having discussed Reputation Systems, we have pointed out that a conventional Reputation System is unlikely to work in a P3P environment mainly because of the lack of feedback. As a result we have proposed an alternative to the conventional approach in the form of a trusted third party. Feedback, although not immediate, is implemented as a function of time while confirmed reports from users of a Web site are dealt with in the same way. Furthermore, reputations of Web sites in the system that we have proposed do not start off with exceptional reputations. Instead a new reputation is given an average rating and then encouraged to improve over time.

We discussed an alternative method to visualising reputation by compar-

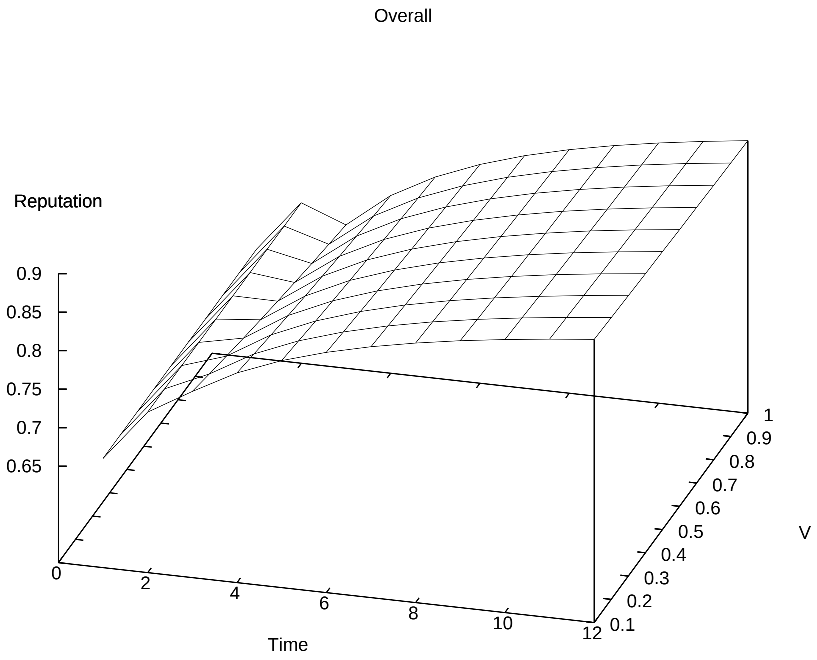

Figure 5.4: The effect of v ranging from 0.1 to 1.0 using a 12 month reputation with a single confirmed report against it.

ing a reputation to the cross section of a tree. We suggested a simple formula that may serve as a guideline towards building up a reputation as a function of time. Possible future work may involve investigating the significance of v (the desired impact of a confirmed report on a user's negative feedback).

From the previous sections, it is clear that without a Reputation System of some kind, trust between a user and a Web site within the context of P3P will have to have been earned through some other means. In this work, the Reputation System we have proposed allows users to gauge the trustworthyness of a site. For search engines that have exposed a P3P policy and earned a good reputation, their users have all they need to make informed decisions as to whether or not they adhere to their own privacy requirements.

5.4 Proxies and P3P

Although the way in which users and their associated P3P agents interact with a P3P compliant Web site has received much attention, the surreptitious role played by the Web proxy server and its impact on P3P has been neglected. In this section we consider the scenario of a Web proxy that is situated between a Web user and a P3P compliant Web site.

In focusing on the role played by the Web proxy from a P3P perspective we will discuss why it is imperative that the Web proxy is identified as a possible privacy threat. In doing so, it will be made clear that the Web proxy must not be excluded in so far as providing a P3P policy to the Web user. The P3P policy provided by a proxy however, brings with it a new set of problems. We analyse these problems and discuss potential solutions.

Transparent proxies are included in our analysis of proxies as potential privacy threats primarily because their nature is somewhat different to that of regular proxies and therefore deserve special attention. A transparent proxy functions like any other proxy with the exception that it is not explicitly configured by a user. Usage of a transparent proxy is in most cases configured by a network administrator on behalf of a user. As a result, although a user may have configured his Agent to make use of a trusted proxy, usage of the trusted proxy may be through an unknown, untrusted and undetected proxy, i.e., the transparent proxy.

The problem presented by chained proxies presents as much of a threat to privacy as transparent proxies and is therefore also included in this thesis. The chained proxy scenario arises from one proxy acting as a client to another proxy. A user may trust proxy P_1 and is content to access P3P services through proxy P_1. As was the case with transparent proxies though, the user may be implicitly going through yet another proxy which he does not know or trust (in the case of proxy P_1 being configured to use proxy P_2 in order to access the Web).

In this section, we focus on why the Web proxy must be considered a threat to privacy. We then discuss the problems introduced by a proxy in more detail as well as possible solutions. This includes a discussion of chained proxies, transparent proxies and the semantics of P3P from a proxy perspective.

5.4.1 Dealing with Web proxies in P3P

The P3P 1.1 Specification [31] only recognises proxies as a cache that may be holding P3P policies belonging to the Web site that a user wishes to access. There is no discussion as to how Web proxies should implement P3P or the way in which user agents should work with proxies that implement P3P.

Consider the P3P policy of section 5.2. This policy essentially states that users of the site will not be logged, i.e., all identifiable data (including the IP address) will be anonymised. If user Bob has specified that he does not want his IP address logged when accessing a Web site, this policy will meet his requirements.

In order to consider the issues that arise when adding a proxy to P3P we include a P3P proxy in the scenario above. For the moment, a P3P proxy is merely a Web proxy with a P3P policy defined for itself, i.e., a P3P compliant proxy.

To minimise the impact of introducing a P3P proxy, the associated P3P policy is accessed via a Policy Reference File which would in turn be accessed via any of the methods indicated in a previous section. The P3P proxy we will use in this example is configured to log all Web access attempts (including the time and IP Address) indefinitely.

In accessing the Web site in question, Bob's P3P agent realizes that Bob's browser is configured to make use of a P3P Web proxy (note that this is not a transparent proxy) which will then access the Web site on Bob's behalf. Bob's P3P agent must therefore scrutinize the policy of the P3P Web proxy to ensure that Bob's privacy is not being violated before making use of any other service on the Web.

Since the Web proxy's configuration stores identifiable information indefinitely, further usage of the proxy will result in a direct conflict with Bob's privacy preferences. If Bob is serious about not wanting his IP Address logged then he will not be able to access any Web sites at all (regardless of whether or not the P3P policies on the Web sites he will be accessing are acceptable).

The consequence of introducing a third party into the P3P framework may have dire implications since P3P was designed with only two parties in mind:

1. A user on the Web with privacy preferences employing the services of

2. a Web site with a P3P policy on a Web server.

The addition of a P3P Web proxy cannot be regarded as just an entity that needs to be scrutinized before accessing the Web. The nature of the proxy is such that there are many opportunities to violate the privacy preferences of an individual who uses it. For example, aside from the logging implications of a proxy, simply caching an object may result in a violation of privacy. A cached object means an attacker using the proxy will be able to determine whether or not a particular object on the Web has been accessed by another user of the proxy, perhaps opening the door to an inference attack of some kind.

Note that Web proxies issue requests to the Web for more than just one user (consider work or academic environments). Though many users may have similar privacy preferences, it is unlikely that their preferences will be exactly the same. If it is the will of the Web proxy administrator to respect the privacy of each of the users accessing the proxy then the administrator will either have to configure a policy that satisfies all users (whether this is feasible remains to be seen) or configure separate policies for each user (adding significant workload to the proxy).

Alternatively, a user could choose to make use of multiple proxies depending on his privacy expectations. It may be the case that proxy P_1 has a different privacy policy to proxy P_2. A user could then decide to employ the services of proxy P_1 for some sites and proxy P_2 for others. Though this seems like a viable solution, this does not apply to users that have only one proxy as a point of contact with the Web (in the case of administrators not agreeing to configure multiple proxies).

5.4.2 P3P Web Proxy Problems

By including the Web proxy in P3P, several problems arise that demand immediate attention. We discuss these problems and examine ways in which they may be resolved. Briefly, the problems are as follows:

> *The semantics of P3P policies.* As mentioned previously, the P3P framework was designed to address the privacy needs of a user accessing a Web site. The addition of a third party into the framework (the Web proxy) will result in subtle changes to the semantics of P3P policies.
>
> *Complicated proxy policies.* Satisfying the privacy needs of each user using a proxy may result in incredibly complex proxy policies. Since configuring these policies will be an arduous task, we must investigate a simpler alternative.
>
> *Transparent and chained proxies.* Transparent proxies are not explicitly defined to be used by the Web user. They may be the result of a sophisticated networking architecture and are therefore unseen by the Web user. Chained proxies occur when a single proxy acts as a client to a second proxy, this proxy will then issue a request to yet another proxy (or the Web) on behalf of the client proxy. What must be investigated is

how these proxies will identify themselves to the Web user and whether or not this identification is necessary.

In considering and discussing these problems it will be evident that Web proxies cannot be ignored within the P3P architecture.

P3P Semantics

The central theme of P3P centers around a client and a Web site. The Web site provides a policy, after perusal of the policy the client may decide whether or not to continue usage of the Web site.

In communicating with a proxy, a user is indirectly communicating with a Web site. As a result, the privacy policy of the proxy as well as the Web site must be acceptable to the user before any indirect communication between the user and the Web site can begin.

A simple solution to this problem is to have the proxy agree to implement the privacy policy of the site that is being accessed through it. If a user is content with the P3P policy of the site then it stands to reason that he should be content with the same policy being applied on the proxy that he is using to access the site.

Whilst this solution is at first attractive, it has the disadvantage of the proxy not having a say in the policy that is to be implemented. It may be the case that the proxy does not employ the same privacy practices at all and even if it was willing to change its practices for the duration of the session, the overhead in doing so for multiple sessions and many users may be far too much to deal with.

The solution proposed in this work allows for a proxy to specify separate policies for individual sites. Because of the indirect means of communication between different Web sites and multiple users, a proxy must be able to provide more than one single policy which describes how it deals with all data collected. It must be able to accommodate for detailing how it intends to handle information with regards to separate entities (Web sites) on the Web.

This is best explained with another example: although Bob has no problems in having his details logged (both at the proxy and at the Web site) when he accesses an online weather service, he is not willing to compromise any privacy at all in so far as electronic payments or online banking is concerned. On the one hand he does not mind the proxy keeping logs for *generic* Web access, but on the other hand he does not want any details logged at all for what he deems as *private* and *confidential*.

In order to support multiple policies we propose changes to the Policy Reference File (the `<POLICY-REF>` element in particular). The Policy Reference File, as mentioned earlier, refers to a P3P policy (or policies) and describes various attributes regarding the policy. We propose the addition of the **siteURI** parameter (as detailed in Table 5.1) which will typically denote the entity (Web site) for which the policy being referred to will be applied. The addition of this parameter allows a proxy P3P policy to specify which URI the P3P policy in question refers to.

```
policy-ref =
  <POLICY-REF
    about="URI-reference"
    [siteURI="URI-reference"]>
      ...
  </POLICY-REF>
```

Table 5.1: An optional parameter extension to the POLICY-REF element.

Absence of this parameter in the `<POLICY-REF>` tag of a P3P proxy policy denotes a *generic* policy, i.e., the policy that applies to all sites accessed via the proxy. Only sites that have defined policies (via the **siteURI** parameter) on the proxy will be excluded from the *generic* policy. Essentially, the **siteURI** parameter provides a mechanism for the proxy Policy Reference Files to indicate which P3P policies apply to certain URIs.

Figure 5.5 illustrates the process of a proxy looking up the appropriate policy to apply for the user in question. This is based on the URI of the site being accessed by the user. The process is as follows:

Step 1 - Client sends a request to the proxy to access a Web site on its behalf.

Step 2 - The proxy looks up the policy associated with the Web site being accessed. The proxy may choose to categorise P3P policies per user in addition to the siteURI tag.

Step 3 - If there is a P3P policy that applies to this user for the Web site in question, it is loaded by the proxy. If there is no applicable P3P policy then the generic policy (depicted as * in the figure) is loaded for the duration of the transaction with the user.

Step 4 - The response from the Web server is forwarded to the client.

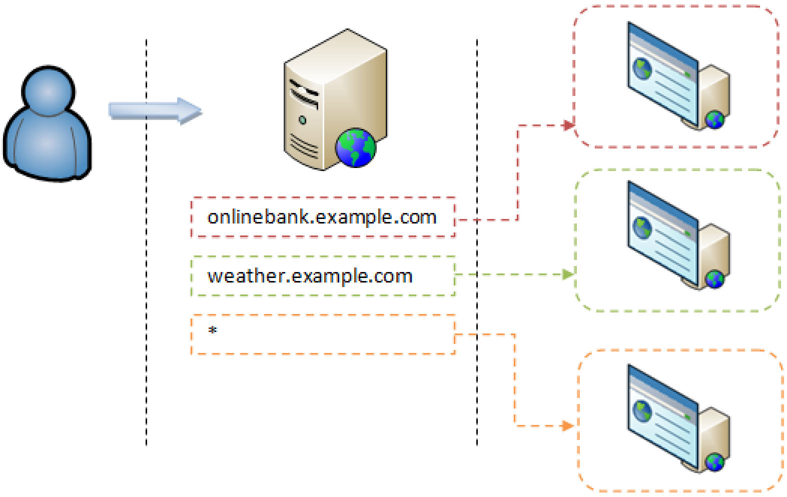

Figure 5.5: User accesses a Web site through a P3P compliant Proxy. If applicable, the proxy will load and adhere to a policy specific to the site being accessed.

The practical implications for proxies that choose to make use of the **siteURI** parameter may be catastrophic. Three problems are immediately apparent:

- The proxy may be subjected to additional stress as P3P policies are requested by users for sites that they may wish to access through the proxy.
- There is additional overhead on the entire process of requesting an object from a Web site as the P3P policy of the proxy for a site is looked up by the proxy upon each initial request to a Web site.
- If an administrator can be convinced that it is imperative for the proxy to strive towards meeting the privacy expectations of all its users then there may be considerable complexity in implementing generic proxy policies for the users, or alternatively, implementing different policies (possibly for different users) for different sites that are accessed via the proxy.

The first problem has already been anticipated in the P3P framework and is circumvented via a simple leasing scheme through the `EXPIRY` element in the Policy Reference File. The first time a user requests a policy from the proxy for a site then the user, in looking up the EXPIRY element, may gauge how long the policy will be valid for. This approach reduces the overhead of always requesting the policy from the proxy for a site each and every single time the user wishes to access the site.

A caching solution may help to alleviate stress on the proxy in so far as addressing the second problem i.e. a proxy can cache policies for quick retrieval at a later stage. Unfortunately, there may still be significant overhead in the initial lookup of a policy.

The third problem is not as easily tackled. We discuss this problem in detail in the next section.

Complicated vs Customised Proxy Policies

A silver bullet policy that addresses the privacy needs of all individuals making use of one proxy will be a rare feat since privacy preferences of users across the Web obviously differ tremendously. Attempting to create a single privacy policy for a P3P proxy that is a cross section of most users' privacy preferences will be an arduous task that, in most cases, will surely fail. The introduction of the **siteURI** tag into the <POLICY-REF> element alleviates the problem only slightly since it allows for proxies to simplify their policies by using multiple policies for different sites.

Needless to say, there is room for improvement in addressing complex privacy policies. Large proxies may have to implement a number of privacy policies to cater for the growing demands of their users. Unfortunately, this solution may not be feasible at all for larger proxies when one considers the sheer number of new sites that may be accessed by hundreds of thousands of users every day.

Simpler schemes may involve generic policies that are served to the general population of a proxy and customised policies that are served to a handful of users either because they pay for the customised policy or maybe because they simply don't fit the generic mould.

5.4.3 Transparent and Chained Proxies

In order to make an informed decision in the P3P architecture, a Web user needs to be aware of all elements that may be a threat to his privacy. We have identified the proxy as a potential threat to the Web user's privacy. A proxy must implement a P3P privacy policy detailing its intentions to the Web user. Transparent and chained proxies introduce another problem to the P3P architecture. In the event of a transparent proxy being present, the user may not be aware of its existence therefore any decision made by the user with regards to his privacy is not an entirely informed decision. It must therefore be the responsibility of a transparent proxy (of any Web proxy) to identify itself as a proxy to the P3P Agent of a Web user.

One mechanism that could be used for identification is that of the proxy detecting when a P3P policy of a site is accessed through it and injecting its own policy into the policy of the site that is sent back to the user. This approach has two benefits. The first is that of the actual identification and the second is that of having saved the user a trip to the proxy to lookup the appropriate proxy policy. A disadvantage of this approach is the overhead incurred in having to monitor all traffic accessed through the proxy.

The identification process we propose may take place through any of the means described in the P3P Specification [31]: HTTP Response Headers, embedded link tags, et cetera. In the case of HTTP Response Headers, we propose the addition of an optional field to the P3P header, the `proxy-policy-ref-field`:

```
[proxypolicyref= URI-reference]
```

Table 5.2: Optional addition to the P3P Header.

The addition of the `proxy-policy-ref-field` in the header (inserted by the proxy) will point to the URI of the P3P Policy implemented by the proxy. Not only does the addition of this field serve as a means that can be used by the proxy to identify itself as a part of a Web-based transaction, but it also points to the P3P policy published by the proxy.

In addition to the optional field, we propose the addition of a new element to the P3P vocabulary to enhance privacy policies for P3P proxies. The addition of the `<PROXY>` element as one of the root elements (an element in the `<META>` namespace - the same namespace used by `<POLICY-REF>` namespace) will allow P3P agents to immediately identify the P3P policy as one belonging to a P3P proxy.

In identifying the policy as a policy of a P3P proxy, the `<PROXY>` element will also give details as to any other proxies (or sources) that may be used in retrieving Web objects. These may be chained or transparent proxies. Table 5.3 describes the `<PROXY>` element.

The `<PROXY>` element contains an optional parameter denoting whether or not the proxy itself is a transparent proxy. Elements within the `<PROXY>` element may refer to other proxies that are used by the current proxy. It may be the case that in accessing a site, the current proxy will make use of

```
proxy =
 <PROXY about="URI-reference" [transparent=true/false]>
   [<PROXYSOURCE siteURI="URI-reference" [transparent=true/false]/>]
 </PROXY>
```

Table 5.3: The PROXY element.

another proxy. The user will now be made aware of this process and will be able to query the privacy policy of the other proxy used.

We allow the proxy to specify which proxy will be used when accessing a site by adding another extension to the <POLICY-REF> element. The optional proxySource parameter in the <POLICY-REF> element will typically refer to one of the <PROXYSOURCE> elements defined in the <PROXY> namespace.

```
policy-ref =
 <POLICY-REF
  about="URI-reference"
  [siteURI="URI-reference"]
  [proxySource="URI-reference"]>
    ...
 </POLICY-REF>
```

Table 5.4: An optional parameter extension to the POLICY-REF element.

These simple additions to P3P allow proxies to achieve two important objectives that must be realised in an effort to minimise the impact of proxies as a threat to privacy:

1. Proxies can identify themselves to a Web user, even in the case of the proxy being transparent.

2. Proxies can list any additional proxies that may be used (via the <PROXYSOURCE> element) in addition to when each of these proxies will be used i.e. for which sites they will be used (from within the <POLICY-REF> element).

What must receive further attention in future research is a mechanism for handling policies that are inaccessible to the Web user. Consider the following scenario: proxy P_1 uses proxy P_2 to access a Web site. proxy P_1 has a P3P policy and proxy P_2 has a P3P policy. In accessing proxy P_1, a

Web user is made aware of the chained proxy framework that proxy P_1 is a part of. The user knows that proxy P_1 uses proxy P_2 to access Web site, it is therefore essential that he scrutinizes proxy P_2's privacy policy before making use of any Web sites.

But what will the outcome be if the user is unable to access proxy P_2's privacy policy (possibly due to firewall restrictions)? There may be a solution to this problem by including a cached copy of proxy P_2's policy on proxy P_1. This approach however may have serious performance implications when several proxies are used in succession, which proxy policies should be stored where, and for how long?

5.4.4 Identification and Separation

In discussing the P3P proxy, this section has centered itself around two themes: Identification and Separation. It must be the responsibility of the Web proxy to identify itself to the Web user as a Web proxy. Identification is possible through the proposed optional field in the HTTP Response Header.

Having addressed identification of a proxy, one can begin to address the issues that arise with transparent and chained proxies. We have discussed these types of proxies as well as the problem they present in a P3P environment. A potential solution has been proposed in the form of extending policy files to specify the nature of the proxy being accessed as well as the nature of any additional proxies that it may make use of.

Having identified itself, the proxy enables users to identify policies on the proxy that will be applied when accessing specific sites through it. This is achieved via the **siteURI** parameter of the `<POLICY-REF>` element. Absence of this parameter denotes a generic policy, i.e., a policy that will be applied to all sites without policies.

In recognising the proxy as a privacy threat we have discussed several issues that are of importance when the P3P Web proxy is introduced; in particular, we have discussed issues relating to semantics and complex policies as well as transparent and chained proxies.

This is by no means a complete solution to the P3P proxy problem. Issues that require further attention in future research are as follows:

- Examining whether the burden of complex policies can be addressed as discussed in section 5.4.2. Could a feasible solution lie in the copying of policies or perhaps a simple kind of categorisation process?

- An investigation into the implications of a proxy caching Web objects. It may be in the best interest of the user not to have any Web objects that he has requested cached on the proxy at all since caching copies of a Web object opens the door several types of attacks, including timing attacks [49].

- Firewalls and Gateways pose just as much of a threat to privacy within the context of P3P as Web proxies. Can the strategies proposed in this thesis also be applied to them?

- Since there are several intermediaries involved in a typical Web transaction, the potential for violation of privacy preferences extends beyond the proxy and onto different layers of the Open Systems Interconnection Reference Model (OSI [35]). Though the approach adopted in this thesis may apply to some of the intermediaries (a firewall is also on the application layer) it is not as easily adopted by intermediaries that are further down in the OSI model. Since a router operates on the network layer of the OSI model it is not a candidate for the type of approach presented in this thesis. Future research would do well to investigate if it is acceptable to simply accept several types of intermediaries as static privacy threats.

5.5 Search Privacy and Personal Control

We have looked at two problems with a technology that embodies the personal control category: P3P. The first issue we dealt with was that of trust: how can a user trust the privacy policy of a Web site? The solution proposed in this work was in the form of Reputation Systems. Positive feedback of a reputation is delivered into the system as a function of time and negative feedback in the form of confirmed privacy violations. With a Reputation System in place, future users can make informed decisions when deciding whether or not to trust the policy of a Web site.

The second issue with P3P is that of proxies. Making decisions without taking into account the practices of the underlying proxy means that a user may be making an uninformed decision. We proposed that proxies should themselves implement P3P policies. We highlighted the themes of Identification and Separation. Our solution to dealing with transparent proxies implies that they are willing to surrender themselves (in the form of identifying that they exist). As much as proxies that identify themselves can fall into the Reputation System presented and in time become trusted, a transparent proxy with suspicious privacy practices has no incentive to identify itself (and start earning a bad reputation).

P3P does enhance the control that users have over their privacy when employing the services of legitimate Web sites. With a Reputation System in place and proxies that implement policies this control is greater because decisions made by the users are more informed. If one accepts that there are entities on the Web that are simply beyond one's control (routers, firewalls, transparent proxies), then the solutions offered in this chapter may be sufficient.

In the event that this naive view of privacy is acceptable, we ask the original questions related to search privacy:

- *Does this PET category have the ability to enhance a user's search privacy?*
  The answer to this is simple: absolutely. This category allows for more informed decision making. Not only does a user know more about what will be done with information collected by the Web site in question (the search engine in this case), but the user can use the reputation of the site to determine whether or not its promises are genuine.

- *Is the search engine still a threat to the user's search privacy?*
  Regardless of whether we are accepting a naive view of privacy, there is still a problem that remains in each of the categories we have looked at thus far: the user's queries are being sent to the search engine. Once the search engine has constructed a search profile from these queries, what happens to it is beyond the control of the user (consider for example if the search engine is compromised or if information is accidently sold). With this in mind, the search engine is still a threat to the user's search privacy.

Given that search engines are still a threat (even with a naive view of privacy), we continue the search for a category that can satisfy our definition of search privacy.

Chapter 6

Search Privacy Through Private Communication

"You have no privacy anyway, get over it"
Scott McNealy, CEO Sun Microsystems, 1995

6.1 Introduction

Private communication between two parties can be achieved through encryption and/or obfuscation. The former, if supported by the search engine, is useful since it thwarts eavesdroppers from recording an individual's search profile. Whilst this does enhance control over one's search profile, it does not eliminate the search engine as a threat to one's privacy (since the search engine still receives the queries and therefore has ultimate control over the search profile stored). The latter is the subject of discussion in this chapter. If we consider the search engine as an eavesdropper (an entity we know will intercept all communication) then, since encryption is not an option, we want to determine if we can obfuscate the data in a manner that will eliminate the search engine as a threat to search privacy.

We assume that the search engine will not adhere to the commitments made in its privacy policy (whether this is intended or not is inconsequential). The TrackMeNot (TMN) extension to the Firefox browser makes a similar assumption. The premise of this technology is to generate noise from the client (by sending false search queries to the search engine), eventually resulting in the real search queries being "*hidden*".

We begin this chapter by evaluating this approach as a means of enhancing control over a search profile. Having discussed this briefly in a previous chapter, it is evident that it does enhance a user's search privacy; what remains to be answered is whether or not the search engine is still a threat.

6.2 TrackMeNot

Howe et al [66] refer to TrackMeNot as means of achieving privacy through obfuscation. They describe the technology as a "lightweight Firefox browser extension designed to ensure privacy in Web search by obfuscating a users actual searches amidst a stream of programmatically generated decoy searches". They realise that there may be an effort from the search engines to thwart this technology and as a result have implemented several features that make it difficult to do so.

The first feature is that of Dynamic Query Lists (DQL). This is a list initially seeded by random terms retrieved from a number of RSS feeds upon being installed on a user's machine for the first time. It is then used by TMN as a source for queries sent to the search engine. In time, TMN will mark terms in the DQL for substitution. Results from queries generated from these terms will be scanned for eligible replacements. Essentially, the DQL evolves through time to be unique to the TrackMeNot user in question. As a result, the queries generated by the tool and the search profile eventually stored by the search engine will also be unique to the user.

The second feature is that of Selective Click-Through. In an effort to mimic a normal user, TMN will click-through on (navigate to) some of the links returned from a query (the "more results" link, for example).

The third feature is Real-Time Search Awareness (RTSA). This monitors the user's browser and makes TMN capable of detecting when search queries have been initiated by the user. This serves as the foundation for two additional features: (1) the exact headers used by the actual user can be used in queries sent by TMN and (2) instead of randomly spaced search queries, TMN can issue a number of queries within close proximity to the user's actual search queries.

6.3 Recognising TMN

The authors of TMN have created a tool which mimics search queries submitted by humans. With such a rich feature set, one can easily argue (and the authors do) that it will be frustrating for the search engines to distinguish between TMN generated queries and actual user's queries.

From a user's perspective, TMN provides an enhancement of control over one's search profile in the form of obfuscation. In addition to legitimate search queries submitted by the user, TMN generates a number of false queries and submits these on the user's behalf. The search profile stored by the search engine is therefore a result of a number of legitimate queries hidden in and amongst a sea of false queries.

From a search engine's perspective, not only are their resources being consumed by false queries, but the profiles stored no longer serve as extremely rich sources for research and data mining. We believe these two reasons serve as a strong motive for a search engine to want to be in a position to distinguish between queries that are legitimate and queries that have been falsely generated.

Distinguishing between these types of queries is essentially a matter of determining whether or not the entity behind the query is a machine. Search engines already have incredible financial interest in solving this problem, specifically for the purpose of thwarting click fraud (as discussed in chapter 4). Since dealing with click fraud is related to solving the problem of distinguishing machine generated activity from that of a human, it seems plausible that this technology can be applied to thwarting TMN.

The following two scenarios are simple examples of the technology that can be used to detect queries generated by TMN:

Invisible content - the search engine provides results containing links that cannot be seen by a human, i.e., they are not visible on the screen and can only be *seen* by the machine. For example, the colour of the link could be the same colour of the background. As legitimate invisible links, the search engine knows that when they are navigated to then the associated search query from which it was served was probably generated by a machine. Of course, this example does not take screen reading software into account.

Tracking users - Javascript is a common technology on the Web which can be used to serve as a foundation for frameworks that track and monitor user behaviour when visiting a Web site [16, 15]. Previous research has focused on tracking the mouse movement of a user in order to determine their search query intent [62]. We believe a similar approach can be used to determine if a user on the Web site is human by concentrating on two simple characteristics: (1) when clicking on a link, is the position of the click always in the same place and (2) when moving the mouse from one position to another, is it always in straight lines?

This is by no means a comprehensive list of the techniques that can be used to determine if a search query has been generated by a human. We highlight these techniques only to emphasize that search, or the act of searching on the Web of today, has become an integral part of our lives that is far more complex than just the process of sending text queries through to a search engine. Whilst it is not impossible to mimic the behaviour of a human in the scenarios above, perfecting this type of technology (simulating the activities of a human) is far beyond the scope of what the authors who proposed the search privacy through obfuscation technique had originally intended.

6.4 Obfuscation and Search

As we highlighted earlier, private communication between a sender and a recipient can be achieved through encryption and/or obfuscation. Within the context of search, encryption would protect queries sent to the search engine from eavesdropping but it would not protect the query from the search engine itself. Since the search engine receives and processes the query it is able to construct a search profile and is therefore still considered a threat to search privacy. One could encrypt the query so that the search engine would be unable to see it but, for obvious reasons, this would be counterproductive.

The second option is that of obfuscation. The authors of TMN chose to use an obfuscation technique that hides real queries to the search engine amongst a number of falsely generated queries. Unlike steganography, where one tries to conceal the very existence of the message exchanged between a sender and recipient [72], this technique generates noise around the legitimate queries sent to a search engine (the intended recipient) resulting in an inaccurate search profile stored. This method of obfuscation extends more control into the hands of the searcher since he/she is able to indirectly influence what it is eventually stored by the search engine. The increase of control can be said to enhance privacy since, as discussed earlier, privacy is directly related to the control one has over one's search profile.

Unfortunately, the search engine remains a threat to one's privacy if it is able to differentiate between queries that have been falsely generated (by a machine) and queries that are legitimate (sent by a human). In the previous section, we discussed the motive a search engine would have for being in a position to do this and also pointed to previous work that could be applied to solving this problem. Essentially, detecting this form of obfuscation can be achieved by detecting queries that have been generated by a machine. What remains to be discussed is whether or not this obfuscation can be detected if it is not generated by a machine.

In chapter 4 we discussed search privacy through anonymity, i.e., using crowds to conceal one's identity from a search engine. Upon joining a crowd the participant (a jondo) is assigned a random path through the crowd that will relay messages to one another and eventually the intended recipient (the search engine) on the jondo's behalf. We now consider a crowd that has been modified as follows: instead of assigning a jondo a random path upon joining the crowd, a path is assigned each time a legitimate query is issued. The jondo chosen to be the exit point from the crowd will exhibit similar behaviour to the user that initiated the search (similar to the TMN features discussed earlier).

For example, user A joins the crowd and starts using search engine X. Queries from user A to X may be routed through the crowd and eventually through user B, who was also using search engine X at the time. In this scenario, the false queries are still routed through to a user's search profile. The difference is that the entity behind the false queries is not a machine.

In this scenario, the problem of determining a user that is obfuscating his/her search profile is no longer related to detecting queries generated by a machine. A jondo in the crowd will essentially have its legitimate queries hidden amongst a sea of legitimate queries that have been generated by other jondos (humans in the crowds).

Unfortunately, this does not mean that we have finally found search privacy. The problem of detecting queries generated by a machine has been shifted to detecting queries generated by other humans, surely a much more complex problem. Whilst this does offer promise in so far as obfuscation techniques it has also introduced another problem: that of trying to achieve privacy through anonymity. We concentrated on this problem in chapter 4 and solutions in this category were shown to be unsatisfactory in so far as eliminating the search engine as a threat to one's search privacy.

6.5 Conclusion

In this chapter, we have investigated the category of private communication as a means to achieving search privacy. Having identified encryption as an unsuitable approach within the context of search, we turned to obfuscation. Essentially, we want to hide information from the very entity we are sending it to.

The TrackMeNot extension set out to achieve this by hiding legitimate queries amongst a number of falsely generated queries. The authors of TMN implemented an array of rich features which make detection of the false queries extremely difficult.

Instead of focusing on the queries themselves, we believe that the search engine is still a threat if it chooses to focus on the activities behind the queries and not just the queries themselves. A number of advances on the Web make this a feasible initiative. Furthermore, we discussed a motive for the search engines wanting to be in a position to detect these activities and pointed to current initiatives that could be applied to this effort (thwarting click fraud).

With a motive and the ability to differentiate the artificially generated queries from the genuine ones, the search engine can "deobfuscate" the search profile. We then discussed a modification to crowds that would have users with similar search behaviours submitting queries on behalf of each other. The difference in this scenario is that a human being would be behind each *false* query (essentially thwarting the deobfuscation technique proposed – machine vs. human activity). Unfortunately, this technique depends on search privacy through anonymity (something we have already shown is not viable).

Since an accurate search profile can be maintained regardless of a user's attempts to obfuscate it (be it via machines using tools like TMN or through human activity in the crowds scenario), the search engine remains a threat to search privacy and our search for search privacy continues.

Chapter 7

A Search Network

"By believing passionately in something that still does not exist, we create it. The nonexistent is whatever we have not sufficiently desired. "
Nikos Kazantzakis

7.1 Introduction

We have evaluated each of the three remaining PET categories that showed potential to deliver search privacy. Whilst all three do show promise in so far as enhancing the control one has over one's search profile, neither has eliminated the search engine as a threat to search privacy.

In chapter 3 we discussed the privacy violation that occurred when AOL released a portion of their search logs to the public. In trying to determine the point at which the privacy violation occurred, we concluded that since the emphasis of search privacy is control over one's own information then search privacy was lost the moment each user started searching on AOL's search engine. Once the user issues queries to the search engine, a search profile can be constructed and controlled exclusively by the search engine. Since control of this profile lies in the hands of the search engine, the search user has lost search privacy.

This does not mean search and search privacy are mutually exclusive. Consider the following:

- We have a search query
- We want a search result

We could submit the query to a search engine, but this would eventually lead to a violation of search privacy. If we move the context of this scenario away from search and consider it from the perspective of the Web alone, then we have a query (Web request) for which we want a result (Web response). To alleviate bandwidth contention, Web proxies often facilitate this scenario. One Web request is essentially submitted on behalf of a first caller. The Web response is then cached and served for subsequent callers of the same request. If we can apply this to search in a manner that does not result in the violation of search privacy at the proxy, then the user would be spared from submitting all search queries to the search engine and search privacy would be a reality.

In this chapter we use the database involved in the AOL privacy violation of 2006 to confirm that the search queries of users are not necessarily always mutually exclusive, i.e., users are sharing queries. With this in mind, we outline the basic architecture of a search network with the primary objective of providing search privacy in a manner that is fast and scalable.

7.2 A Case for Sharing

In their analysis of the findings from nine Web search studies, Jansen and Spink [71] point out that the results of one search engine study "cannot be applied wholesale to all search engines." In this section, we confirm the findings of Markatos [83] from the analysis of the Excite search engine logs: search queries have a significant amount of temporal locality (shared queries).

7.2.1 Analysis of Search Data

Of the 650,000 search profiles available in the AOL database, we picked a random sample. This was made up of 65,517 profiles with a combined total of 3,558,412 queries (t). Figure 7.1 depicts the number of times that each of the top 1,000 queries was used.

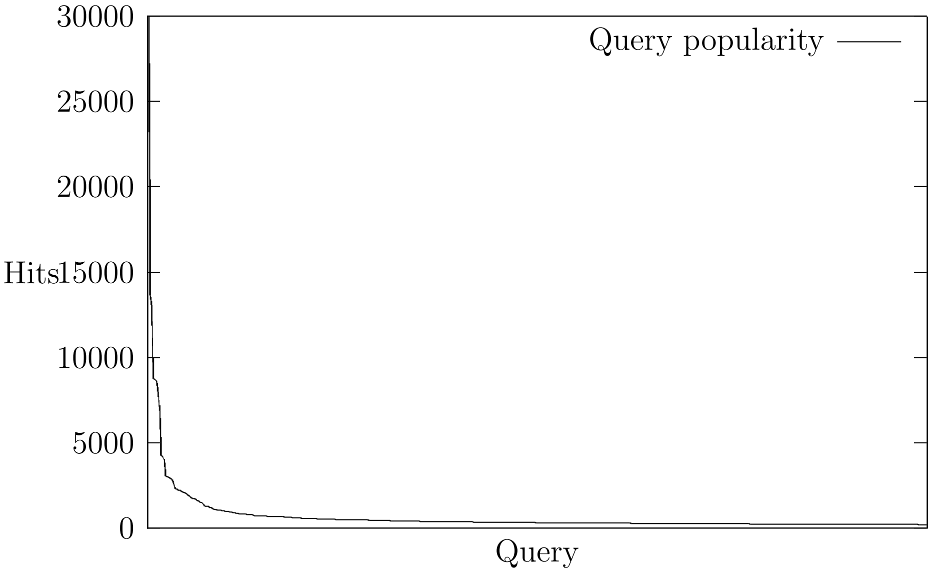

Figure 7.1: The top 1,000 queries ranked in descending order.

A unique query (Q_u) is a query that was issued only once in the sample. A shared query (Q_s) is defined as a query that has been issued more than once by any user/profile in the sample. Query 1 (Q_1) is said to be unique if it has only been issued once by a single user in the system. Q_2 on the other hand, is said to be shared if user x has used it multiple times, or if user x and user y have each used it once or more times. The rapid fall and long tail depicted in figure 7.1 hints at there being far more unique queries than shared. Further analysis of the data shows that this is indeed the case: $n(Q_si) < n(Q_uj)$ where Q_si is the set consisting of all shared queries, Q_uj the set of all unique queries and n the function returning the number of elements in a set. Specifically $n(Q_si) = 479,688$ and $n(Q_uj) = 736,967$.

Figure 7.2 depicts the relationship between users and the types of queries that they have submitted. Each user has two points of interest: (1) the number of shared queries he/she issued in black ($n(Q_s i)$ for the user) and (2) the number of unique queries he/she issued in gray ($n(Q_u j)$ for the user).

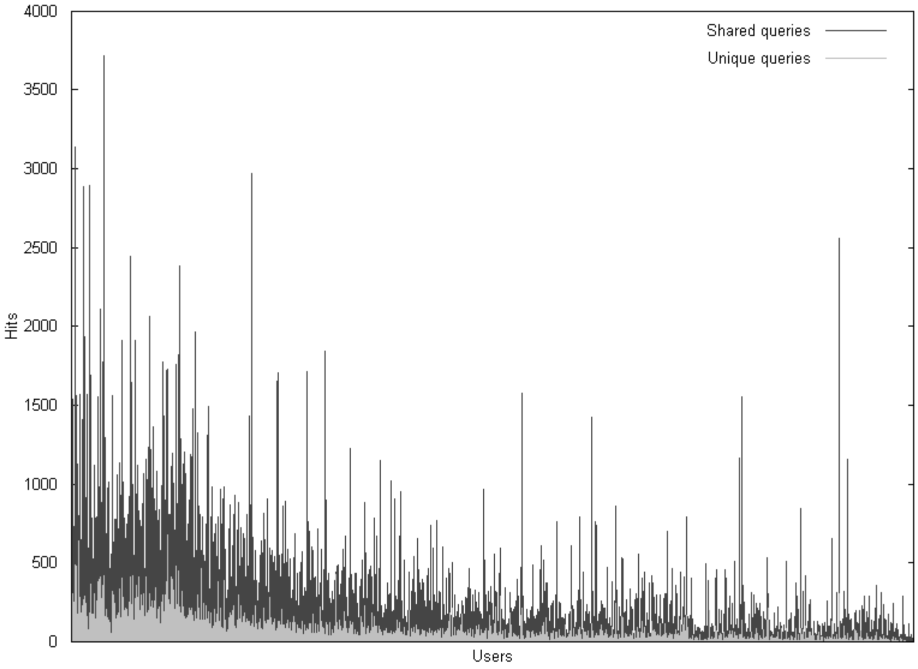

Figure 7.2: Each user is plotted against the number of shared and unique queries that he/she submitted.

Despite the apparent contradiction, the large amount of shared queries is consistent with our previous observation of $n(Q_s i) < n(Q_u j)$. This may not seem initially evident but must certainly be the case since we know that there were a total of approximately 3.5 million queries (t) issued. If 736,967 of those queries were unique then the remaining $Q_s i$ queries must make up the difference. What is now clear is that although $n(Q_u j)$ is almost double the amount of $n(Q_s i)$, if each Q_s were treated as a separate query, then shared queries are far more popular than unique queries: $n(Q_s i) = t - n(Q_u j) = 2,821,445$.

It may be the case however that each user is simply reissuing their own queries. If this is so then a network which depends on sharing query results amongst its users would be useless since there would not be many queries to share.

Figure 7.3 depicts the number of different users that used each query.

The figure shows that at least 50% of the shared queries were used by two or more different users: $0.5 * n(Q_s i) = 1,410,723$. Since this is almost half of the total number of queries issued, we believe this makes an excellent case for a network dependent on sharing search queries.

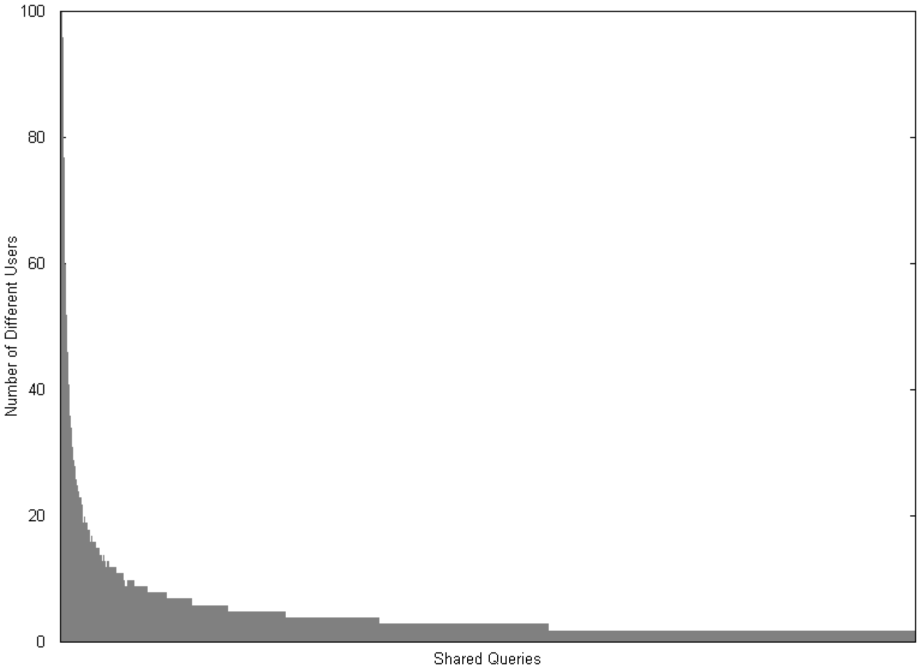

Figure 7.3: In this figure the number of users that used each shared query is plotted.

7.3 A Search Network

We have shown that there is a surprisingly large number of shared search queries. The network we propose leverages off of this fact in a bid to better protect the privacy of its users. In this section, we discuss a high level design of the network. This is achieved through an analysis of its requirements:

1. The network must be scalable. The number of search queries per day for a single search engine may be in the hundreds of millions [25]. A network which aims to operate as a source of search query results for multiple search engines across the Web must scale to accommodate a staggering number of users and queries.

2. The network must be fast. A search query conducted through the network should be comparable to querying the search engine directly.

3. The network must protect the privacy of its users not only from the search engines, but from each other. We must assume that not all users of the network are trustworthy.

7.3.1 A Scalable Network

A centralised search network would not lend itself well to massive scalability unless tremendous costs were incurred. From a privacy perspective, this approach would also mean a single point of vulnerability. A decentralised approach on the other hand, may be far more cost effective in addition to not forcing users to trust a single entity.

A decentralised and scalable approach is that of P2P networking. There are a number of P2P networks which exhibit scalability [2, 127, 105, 113, 54]. Most of these networks are based on the notion of a Distributed Hash Table (DHT) [11].

A DHT is essentially a decentralised approach to storing and retrieving data. Load in a DHT is distributed in a manner that facilitates change in the set of participants with minimal impact to the network itself. Participants in a DHT employ a consistent hash function [74] so as to establish a keyspace partitioning scheme amongst themselves that balances the load fairly.

From a very high level perspective, a DHT offers two sets of functions:

1. $put(k, data)$: k represents a hash of the key used to index $data$ into the DHT. When putting data into the DHT, the key and the data will *hop* through nodes of the DHT until the node responsible for that key is reached (this is determined by the globally known hashing function), it is this node where the data will be stored.

2. $get(k)$: returns the $data$ indexed by k.

If we briefly examine the search process, a search network built upon a DHT seems to be a logical fit: a user issues a query ($get(k)$) to a search engine which responds with a search results page ($data$). For the sake of simplicity, we only store the first page of search results. Note that Jansen and Spink [71] highlight the trend of more search users viewing only the first page returned from a search query. This finding was initially highlighted by Beitzel et al [21] in their analysis of a week's worth of search query logs from approximately fifty million AOL users.

The network we propose would act as a scalable storage depot operating in conjunction with the basic search process. Once a search results page has been retrieved from the search engine it would be stored in the DHT ($put(k, data)$) so as to save other users from having to query the search engine in the future.

7.3.2 A Fast Network

Besides the benefit of being scalable, DHTs offer the advantage of only needing a small number of hops through neighbouring nodes in order to satisfy requests. The order differs across varying implementations of DHTs but it is typically $O(log(n))$ or better (where n is the number of nodes participating in the DHT). Unfortunately, a disadvantage of the quick lookup scheme using the hashing function means that only exact matches for k can be returned.

7.3.3 A Privacy Preserving Network

As the reader has probably inferred from the discussion thus far, the proposed search network works as follows:

- Results of queries submitted to a search engine are stored in a DHT.
- Users wishing to make the same queries (this has been shown to be a likely occurrence) have the option of retrieving the results from the DHT instead of making a submission to a search engine.

At this point, the contribution of the network is that of disconnecting searchers from the search engine. We assume that search queries and their associated results are not personalised. This assumption means that there is a one to many relationship between a result for a query and the number of consumers of the result, i.e., a result for one user can be provided as a result for another user submitting the same query.

Unfortunately, this disconnection from the search engine is not the only concern. Since there may be users of the network that cannot be trusted, the proposed solution must provide all users with a degree of privacy from each other. In a bid to maintain the scalability and speed of the search network, we adopt a simple approach.

Since the users will be relying on each other to retrieve and store results, anonymity within the network is not our goal, i.e., the network does not employ any anonymisation techniques in so far as *who* is searching for data. Privacy is provided in the sense of *what* is being searched for. We know that the DHT stores a search result ($data$) for the hashed version of a query (k). At the very least, we will not store the query or the search result in plain text. Since k is a result of a one way hashing function this only leaves the problem of ensuring that *data* cannot be observed. In the absence of a centralised authority or third party we propose an encryption scheme that uses the plain text version of the query (q before it is hashed) as the key when encrypting the result: $data = E_q(result)$. Since q will form part of the one way hash, the search results will only be accessible to those who already know what they are looking for, in other words, they know q.

At this stage, we have only dealt with storing the results of a query in the DHT. We have discussed a solution where participants of the network are disconnected from the search engines and are offered a degree of privacy from each other. Unfortunately, the privacy introduced in this section brings with it another problem: if the query is hashed and unobservable in plain text, then who is going to submit the query on behalf of a user when a cached result does not exist in the DHT? After all, the search engine cannot be queried with a hashed version of the query.

7.3.4 Submitting a Query

Submitting a lone query to a search engine is not as much of a problem as submitting a number of queries over time since a search engine may construct a sequential log (search profile) of the user's queries which may be used to infer something private about him or her.

A search profile consisting of other users' queries contributes to the privacy of the user in question since each query has lost its context. In order to submit queries to a search engine so that they are inherently stored this way, users make submissions on behalf of each other. Whilst this approach seems to be the simplest solution it has an obvious disadvantage: if user U_2 is to submit query Q_1 to a search engine on behalf of U_1, then U_2 must know what Q_1 is. This may be considered a violation of privacy. However, if our goal is to prevent the search engine from violating privacy, then it could be the case that this potential for violation of privacy may be acceptable in the event that not all queries from U_1 are submitted by U_2. This would have U_2 only submitting ad hoc queries for U_1 and, just like the search engine, these queries would have no context.

Since a DHT is essentially treated as a decentralised collection of nodes used to store data, we introduce a rule that each node must obey in order to have users successfully submitting queries on behalf of each other. The rule is simple: if a predefined data structure is present in the data to be stored, the necessary elements required to make a submission to a search engine must be extracted from this data. The search engine is then queried, the result of which is encrypted and stored in place of the original data.

The data structure serves only as a means for nodes to recognise when to store the data and when to use the contents thereof to formulate a query to a search engine. For example, *data* containing "**BING,Anonymity**" may signal a node to send the query "Anonymity" to the bing.com search engine. The result would then be encrypted (using "Anonymity" as the key) and stored. Note that the proposed search network is not bound to a specific search engine.

Of course, a large number of colluding nodes could possibly thwart the search network's attempt to preserve the privacy of users from each other. Increasing the number of attackers in the DHT increases the probability of receiving a number of queries on behalf of the user under attack and, in doing so, increase the likelihood of generating a log with context. There are a number of options for this scenario. The first option is to bypass the DHT nodes entirely and have a foreign node join the DHT to submit the request on the user's behalf. The foreign node could be part of another PET, for example, Tor [42]. Unfortunately, making use of another network will most likely introduce one of the very problems the search network proposed in this chapter tries to remedy: scalability and performance.

Another option is a variation on the technique employed by "John Doe" nodes in the Crowds model [100]. In this model, a John Doe picks a random node from the crowd (which could be itself) and forwards the message to the node. This node, upon receiving the message, flips a biased coin to determine whether to send the message to another John Doe node or to deliver the message to its recipient. Nodes in the search network could employ a simpler process: upon determining that a search result is not available for a particular query, a node can flip a coin to determine whether to issue a request into the DHT to have the query addressed (using the predefined data structure) or address the query itself and place the result into the DHT. Since the node is part of the DHT it will have already conducted a number of queries on behalf of other nodes. As a result, detecting the user's queries from a search engine's perspective will be cumbersome. Ultimately, the queries still have no context.

We have opted to adopt the latter approach when submitting queries in the proposed search network.

7.4 Formalisation

In this section, we highlight the relationship between an accurate search profile and a log of queries submitted by a user. We then analyse a number of methods that can be used to submit queries to a search engine. The focus of our analysis deals with the logs that can be derived by an entity the likes of a proxy or search engine. We will show that constructing an accurate search profile of a user participating in a search network requires collusion with all parties in the network.

A log of search queries can be prepared from any one of the following perspectives: the user issuing the queries, a proxy user issuing the queries or the search engine for which the queries are destined. In this section we place emphasis on the number of queries in a log. Correct knowledge of a log (and its contents) implies that one knows how many queries have been issued. Similarly, if one does not know how many queries were issued then the contents of the log are not entirely known.

Within the context of this chapter, if a user submits n queries to a search engine, the log of this user from his/her perspective is entirely known and will contain n queries. If the log generated for this user from the perspective of a search engine contains the *same* n queries then the search engine has correct knowledge of the log.

L is a function with the set of users, U, as its domain and the number of queries associated with his/her log, V, as its range. With $U = \{u_1, u_2, \ldots, u_n\}$ and $V = I\!N$, L is the function that maps a user to the number of queries issued.

$$L : U \mapsto V \tag{7.1}$$

For any user u_i, the number of search queries issued is $L(u_i) = n_i$.

Let $L_s(u_i)$ denote the number of queries issued by u_i from the perspective of search engine s. Similarly, $L_{u_j}(u_i)$ denotes the number of queries issued by u_i from the perspective of u_j where $j \neq i$. If $L_s(u_i) \neq L(u_i)$ then the contents of u_i's log are not entirely known by the search engine (it does not have an accurate search profile for u_i).

7.4.1 Submitting Queries Directly

In the absence of a proxy, all search queries issued by u_i are submitted directly to the search engine. Since the number of queries in the log generated from the perspective of the search engine equals the number of queries in the user's log, the search engine has correct knowledge of the log (also referred to as an accurate search profile), i.e., $L_s(u_i) = L(u_i) = n_i$.

7.4.2 Submitting Queries Through a Proxy

Consider the introduction of a proxy where u_j submits queries on behalf of u_i. From the perspective of a search engine, the logs generated yield the following: $L_s(u_j) = n_i$ (there are n_i queries for user u_j) and $L_s(u_i) = 0$ (there were no queries for user u_i). Whilst the search engine can't derive anything about user u_i (there is no log of his/her queries), the user through which u_i is proxying is in a much better position to build a search profile since $L_{u_j}(u_i) = L(u_i) = n_i$. Essentially, the problem for user u_i is that the accurate search profile has now been shifted from the search engine to the user that is acting as a proxy.

7.4.3 Using a Proxy and Direct Submission

We try to alleviate the problem of shifting the profile from one entity to another by introducing the flipped coin approach discussed in the previous section. Instead of forwarding all queries to a proxy we let p denote the probability of forwarding a query to a proxy. Since there are n_i queries submitted by u_i, an average of pn_i of these will be forwarded to the proxy and $(1-p)n_i$ to the search engine. The result is that a number of queries go to u_j and a number to s (note that we are now dealing with expected values):

$$L_{u_j}(u_i) = L_s(u_j) = pn_i \tag{7.2}$$

$$L_s(u_i) = (1-p)n_i \tag{7.3}$$

By splitting the queries between the two entities, we decrease the chance of a single entity forming an accurate search profile.

However, if it is known that u_j is only a proxy for u_i, the search engine can easily construct the complete search profile by combining its log of u_i with the log of u_j. We refer to this log as $L_{s'}(u_i)$:

$$L_{s'}(u_i) = L_s(u_i) + L_s(u_j) \tag{7.4}$$

Using equation 7.2 and 7.3:

$$L_{s'}(u_i) = (1-p)n_i + pn_i = n_i = L(u_i)$$

7.4.4 Using Multiple Proxies and Direct Submission

If we increase the number of users acting as proxies for u_i to m, constructing $L_{s'}(u_i)$ becomes tedious for s since it would have to know each of the m users that u_i is proxying through. We know that the probability of forwarding a query to a proxy is p. With m proxies, the chance of forwarding a query to any particular proxy is $\frac{p}{m}$. As a result

$$L_s(u_j) = \frac{pn_i}{m} \tag{7.5}$$

Compiling the search profile for u_i from the perspective of s would be achieved by adding the logs of all m proxies to $L_s(u_i)$:

$$L_{s'}(u_i) = L_s(u_i) + \sum_{j=1}^{m} L_s(u_j) \tag{7.6}$$

Using equation 7.5, 7.6 expands so that $L(u_i)$ is determined:

$$L_{s'}(u_i) = (1-p)n_i + \sum_{l=1}^{m} \frac{pn_i}{m} = (1-p)n_i + m(\frac{pn_i}{m}) = n_i = L(u_i)$$

As the number of proxies for u_i increases, a successful attack from a search engine perspective becomes difficult only in the sense that it has to know which users are acting as proxies for u_i.

7.4.5 Submission of Queries Through a DHT

In this chapter, we have proposed a network where all users act as proxies for one another. This very simple act adds significant complexity to an attack from a search engine since, as we are going to show, it would have to collude with each of the users proxying for the victim in addition to knowing who is proxying for the victim. For the sake of simplicity, in this section we assume that users in the search network act as non-caching proxies.

Since all users in the proposed network submit queries on behalf of other users in the network, the log of queries submitted by any user from the perspective of the search engine will include all queries submitted by the user directly (remember from 7.3 that this is $(1-p)n_i$) in addition to queries for which the user acted as a proxy. To be precise, in a network of m users:

$$L_s(u_i) = (1-p)n_i + \sum_{j=1}^{m} \frac{pn_j}{m}; i \neq j \tag{7.7}$$

Applying this to 7.6, we have

$$L_{s'}(u_i) = (1-p)n_i + \sum_{j=1}^{m} \frac{pn_j}{m} + \sum_{j=1}^{m} L_s(u_j) \tag{7.8}$$

We have shown that the search engine must take into account that u_i is submitting queries on behalf of other users (depicted in 7.7). Note that each proxy u_i is using is also acting as a proxy to users other than u_i. Furthermore, the proxies are submitting queries themselves. Unless the search engine colludes with each of the proxies (ensuring that they are not submitting any queries themselves), then the log of queries submitted to the search engine from each proxy is similar to that of u_i in 7.7:

$$L_s(u_j) = (1-p)n_j + \sum_{l=1}^{m} \frac{pn_l}{m}; l \neq j \tag{7.9}$$

With this in mind, we expand 7.8:

$$L_s\prime(u_i) = (1-p)n_i + \sum_{j=1}^{m} \frac{pn_j}{m} + \sum_{j=1}^{m} L_s(u_j)$$

$$= (1-p)n_i + \sum_{j=1}^{m} \frac{pn_j}{m} + \sum_{j=1}^{m}((1-p)n_j + \sum_{l=1}^{m} \frac{pn_l}{m})$$

Table 7.1 summarises the logs derived on the search engine and proxy for each of the approaches discussed in this section. Note that in the search network approach, a search engine colluding with a number of proxies will only be successful (generate an accurate search profile) when the number of proxies colluding in the network equals $m - 1$.

7.5 Conclusion

Using the database of 2006, we took a random sample and confirmed that there is a significantly large number of queries which are shared by users when using search engines. If we assume that the results of these queries are not specific to the user conducting the initial query, then the results from a single query can be shared with other users of the network, sparing them from having to conduct the query themselves. In doing so, the user would be disconnected from the search engine.

The disconnection from the search engine and the sharing of queries forms the basis upon which the search network is proposed. Essentially, the search network acts as a cache of search queries and search results. Built on top of a distributed hash table, the search network allows users or nodes to place and retrieve queries and their associated search results from the cache. A simple encryption scheme using the query to encrypt the search results allows for a degree of privacy between the users of this network. The major contribution is that of removing control from the search engines. In this network, exclusive control over one's search profile is no longer in the hands of a single entity.

DHTs have shown themselves to be fast and massively scalable. These two characteristics are of paramount importance in a network of this nature.

Method	**Log on Proxy**	**Log on Search Engine**
Search engine	0	n_i
Proxy	n_i	0
Search engine and proxy	pn_i	$(1-p)n_i$
Colluding search engine and proxy	pn_i	n_i
Search engine and m proxies	$\frac{pn_i}{m}$	$(1-p)n_i$
Colluding search engine and m proxies	$\frac{pn_i}{m}$	n_i
Search network (m users)	$\frac{pn_i}{m}$	$(1-p)n_i+\sum_{j=1}^{m}\frac{pn_j}{m}; i \neq j$
Search network with colluding search engine and proxies	$\frac{pn_i}{m}$	n_i iff number of colluding proxies $= m-1$, otherwise $(1-p)n_i+\sum_{j=1}^{m}\frac{pn_j}{m}+\sum_{j=1}^{m}((1-p)n_j+\sum_{l=1}^{m}\frac{pn_l}{m})$

Table 7.1: A comparison of the different approaches used when querying a search engine.

A greater number of users will result in a higher probability of a query already being conducted by someone else on the network. If the network cannot scale well, there would not be more incentive to use it over conventional PETs.

Ultimately, the proposed search network is a form of distributed proxy. It differs from conventional privacy preserving proxies the likes of Anonymizer.com because the queries, in addition to not always being in the clear, lack context. This is not the case with a centralised proxy since although one is protected from the search engine, the proxy itself has the potential for privacy violation.

Chapter 8

Conclusion

Si finis bonus est, totum bonum erit – If the end is good, everything will be good

In this work, we looked at the privacy problem within the context of search. Having identified the search engine as an entity that could not be trusted to provide search privacy, the aim of this work was to determine whether or not there were existing technologies that could be trusted to do so. Having found that this was not the case, we investigated an alternative solution.

We began this work by looking for a definition of privacy. After examining several approaches to privacy in existing literature, we adopted Westin's definition: "the claim of individuals, groups or institutions to determine for themselves when, how and to what extent information about them is communicated to others". In this definition, emphasis is on control over one's own information.

We then looked at search and search profiles. Given the nature of privacy violations related to search in the past, we view the search profile of an individual as information belonging to that individual. With this in mind, search privacy is essentially defined as the control that one has over one's search profile.

Participating in the search experience verbatim (using a search engine directly) leaves complete control over one's search profile in the hands of the search engine. By extending control of one's search profile to the search engine, one is effectively relinquishing control and in doing so giving up one's search privacy. As a result, search privacy in this scenario is non existent.

With an understanding of what search privacy means and the knowledge that it does not exist within the common use case scenario offered by the search engines of today, we began our search for search privacy through other means. As we examined the potential for search privacy that each PET category could offer, we asked two questions: (1) does the technology enhance control over one's search profile and (2) is the search engine still a threat to search privacy? Within the context of the definition of search privacy that we have offered, a suitable technology must not only enhance control but it must also eliminate the search engine as a threat to search privacy.

We concluded the following from each of the categories examined:

> **Organisational Safeguards**. This category was quickly dismissed as an option for providing search privacy. The reason for this is because total control lies in the hands of the organisation, i.e., it is up to the search engine to implement internal policies that provide users with control. When control of a search profile has already been lost or willingly relinquished by a user, there can be no privacy and as a result, we did not consider this category any further.

Anonymity. From a privacy perspective, anonymity is control over one's identity. This category was initially quite promising. Although one can greatly enhance one's control over a search profile through anonymity using an anonymity network, there exists a motive for the search engine to identify participants in this network. The motive is that of thwarting click fraud. We discussed and simulated an external attack on an anonymity network that was based on two principles: (1) the user responsible sending messages from the network will not represent himself and (2) each user employs a mechanism for search where related queries can be recognised (cookies for example). The result of the attack confirmed that the search engine may be in a position to expose/identify participants in the network. By identifying members in the anonymity network, the search engine remains in control of their search profile and therefore remains a threat to their search privacy.

Personal Control. In this category, the focus is on ensuring that information of an individual is only used in a way that the individual has deemed appropriate, i.e., the individual has control over the use of his own information. We turned to P3P and asked the same two questions. It was quickly evident that like the other categories, P3P did enhance control but there were problems. The first problem was that of trust. Any entity could easily declare that they would abide to a sterling P3P privacy policy, but how does one know that they will adhere to what has been promised? We offered a solution in the form of an already proven technology in similar scenarios: a reputation system. The second problem was that of proxies. P3P neglects to consider the role played by an entity in the middle of a Web transaction. After highlighting this problem, we provide a solution in the form of subtle changes to P3P that essentially considers the privacy policy of any proxy involved in a Web transaction. Despite having addressed these two issues, there exists a more fundamental problem. The user is still communicating with the search engine and a search profile will still be constructed. Therefore, full control remains in the hands of the search engine. A search engine that adheres to its P3P policy can still be compromised and along with it, the profiles of its users.

Private Communication. Encryption and/or obfuscation are considered as a means to achieve search privacy in this category. We first looked at encryption but eventually dismissed this option for although it will provide the user with a more secure channel when communicating with the search engine, it does not protect the user from the search

> engine itself (a search profile is still constructed and controlled by an entity other than the user). We then considered obfuscation and looked at a tool which uses this technique to protect its users from the search engine. By sending waves of machine generated false queries around the legitimate queries of the user, the idea is to leave the search engine with what is essentially an inaccurate search profile. Since the user has full control over the queries submitted to a search engine, this is a legitimate approach which initially looked to be promising. Although search privacy is enhanced in this scenario we found that the search engine remains a threat to privacy. With a motive to determine who is behind a query (something the search engines have a vested interest in doing so as to thwart click fraud), we discussed the scenario of search engines determining if a human is behind a query or a machine. Whilst the query itself may seem perfectly legitimate, it is the way in which the user/machine interacts with the result that can leave the search engine deciding on whether the user is indeed human. In doing so, the search engine can distinguish between false queries and legitimate queries and ultimately generate an accurate search profile.

Of the three categories which proved to enhance the control one has over one's search profile, none of them eliminated the search engine as a threat to privacy. The fundamental problem in each category is that the original search scenario is being replayed when communicating with the search engine: the user is submitting queries to the search engine (either directly or indirectly – in the case of anonymous networks). In each case, the search engine receives the query and when possible will construct a search profile for the user in question over which it will have ultimate control.

In offering a solution to this problem, we asked if it was always necessary to submit the query to the search engine. By not having the user's query to begin with, the search engine will not have anything from which to construct a profile. We turned to proxies, an already incredibly useful technology on the Web. If we could construct a distributed cache of search queries and their associated results, queries would only be sent to the search engine under specific conditions.

The primary use case of a proxy is that of sharing data. If an entity has requested and received data from a remote source through a proxy, one can save having to make another request to the remote source for the same data by making a copy of the data available to other entities sharing the proxy. We believed that this scenario was applicable within the context of search. For any given search query, why retrieve the search results from the search engine when they may already exist in the distributed cache?

We made a case for this by confirming that there is indeed a significant number of search queries that are shared by users. As a result, a search network of this kind would be very useful. In the event that a search query was not in the cache, we proposed a simple mechanism for users in the network to retrieve the result from the search engine and place it in the cache. Since the cache is distributed, no single user will have complete control over a user's queries. Similarly, in the event that the search engine can identify the original source of queries sent to it, the queries form only a subset of all queries sent by the user in question.

The contribution to search privacy comes in the form of an almost complete disconnect between the search engine and the user conducting a search. Since each of the PETs we examined in previous chapters failed to introduce this disconnect, the original search scenario was maintained because every query eventually resulted in a request to the search engine. As a result, the search engine remains in a position to construct a search profile ergo control of the search profile is lost as is the search privacy of the user.

In chapter 3, we described the typical search scenario as follows: online users issue a query to a search engine which results in a set of links which redirect to Web pages with more information. Note that the network we have proposed to address this scenario in a manner which maintains search privacy did not explicitly deal with the redirection problem. In order for there to be a disconnect between the searcher and the search engine, the redirect through the search engine that has been embedded in a search result can not be honored. If the redirect is honored then the original scenario is maintained for the search engine is still directly involved in the search process – it will be in a position to determine what the context of the redirect is and in doing so construct a search profile. In treating the links like search queries in the DHT though, the problem of redirection can be overcome, i.e., $put(k, data)$ where k is the result in the search results page and $data$ the link after redirecting through the search engine.

8.1 Does this enhance search privacy?

The search network can be said to enhance search privacy through two contributions:

> *Disconnection.* By interrupting the traditional search scenario, ultimate control over a search profile has been shifted from the search engine to the user. In a limited sense, one could argue that control is distributed amongst users of the distributed proxy for, after all, the

search results must come from somewhere and there is no disconnection between the users.

Search Democracy. The search network introduces the notion of free and equal representation of search results on the Web, one in which no single entity has ultimate control over deciding which Web sites are right, which Web sites are wrong and, more importantly, who is searching for them.

8.2 Is the search engine still a threat?

If the search engine has the motive to be considered a threat, there is sufficient reason to believe that it will be a threat. Given the nature of search today, we believe the following two scenarios would serve as enough of a motive:

1. A search network acting as a distributed cache of search queries and search results would be threatening one of the search engine's primary revenue streams: advertising. Reducing the number of queries going through a search engine has a direct impact on the number of adverts that can be displayed. Fewer advertisement impressions will ultimately result in much lower click-throughs to advertisers and inevitably less profit for the search engine.

2. Not knowing exactly what results users have chosen for the majority of queries passed through it (via redirects) may have a dramatic impact on the validity of the results offered by the bigger search engines. A dominant search engine that knows what the majority of the world is searching for (in addition to what the majority of the world deems as most relevant from the results provided) does not want to lose this advantage over other search engines.

For as long as the disconnect between the searcher and the search engine exists though, we believe that the search engine is of little threat to search privacy because it will be unable to construct accurate search profiles. However, given the incentive, the search engine has only to restore the connection to the searcher in order to gain full control of the original search scenario (and search privacy). We see two ways of achieving this:

Providing personalised search results. The search network is based upon sharing results. Since there are a significant amount of shared queries, a distributed cache of shared results seems natural and somewhat evolutionary (much like the introduction of the proxy was to the Web). If the search engine can challenge this simple premise and provide search results in a manner that would make no sense to share them, then the network would have lost the foundation upon which it was built. By serving personalised search results the search engine would reduce the benefit gained by sharing queries and in doing so, restore the connection between the searcher and itself.

Attacking the search network. We know that control of a search profile is in some ways distributed to participants throughout the search network. We have not investigated the feasibility of gaining control over a profile by inserting a number of false participants into the network on behalf of the search engine. Since the search engine has a fairly good idea of what queries are shared and likely to be cached in the network, will it be possible for it to only target those queries in the search network? We hope that the work in this thesis has set the foundation upon which to answer these questions in future research.

Bibliography

[1] M. Abe. Universally Verifiable Mix-Net with Verification Work Independent of the Number of Mix-Servers. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 83(7):1431–1440, 2000.

[2] K. Aberer. P-Grid: A Self-Organizing Access Structure for P2P Information Systems. *Lecture Notes in Computer Science*, 2172:179–194, 2001.

[3] K. Aberer and Z. Despotovic. Managing trust in a peer-2-peer information system. In H. Paques, L. Liu, and D. Grossman, editors, *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM01)*, pages 10 – 317. ACM Press, 2001.

[4] E. Adar. User 4xxxxx9: Anonymizing query logs. In *Workshop on Query Log Analysis at the 16th World Wide Web Conference*, 2007.

[5] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Hippocratic Databases. In *VLDB*, pages 143–154, 2002.

[6] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. An XPath-based preference language for P3P. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 629–639, New York, NY, USA, 2003. ACM.

[7] R. Albert, H. Jeong, and A. Barabási. The diameter of the world wide web. *CoRR*, cond-mat/9907038, 1999.

[8] H. Aljifri and D. S. Navarro. Search engines and privacy. *Computers & Security*, 23(5):379–388, 2004.

[9] R. J. Anderson and F. A. P. Petitcolas. On the limits of steganography. *IEEE Journal of Selected Areas in Communications*, 16:474–481, 1998.

[10] F. Anklesaria, M. McCahill, P. Lindner, D. Johnson, D. Torrey, and B. Albert. The Internet Gopher Protocol (a distributed document search and retrieval protocol). RFC 1436 (Informational), March 1993.

[11] F. Arajo. *Position-Based Distributed Hash Tables.* PhD thesis, Department of Informatics, University of Lisbon, May 2006. DI/FCUL TR-06-7.

[12] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke, and S. Raghavan. Searching the Web. Technical Report 2000-37, Stanford InfoLab, 2000.

[13] S. Asadi and H. R. Jamali. Shifts in Search Engine Development: A Review of Past, Present and Future Trends in Research on Search Engines. *Webology*, 1(2), December 2004.

[14] P. Ashley, S. Hada, G. Karjoth, and M. Schunter. E-P3P privacy policies and privacy authorization. In *WPES '02: Proceedings of the 2002 ACM workshop on Privacy in the Electronic Society*, pages 103–109, New York, NY, USA, 2002. ACM.

[15] R. Atterer and A. Schmidt. Tracking the interaction of users with ajax applications for usability testing. In *CHI*, pages 1347–1350, 2007.

[16] R. Atterer, M. Wnuk, and A. Schmidt. Knowing the user's every move: user activity tracking for website usability evaluation and implicit interaction. In *WWW*, pages 203–212, 2006.

[17] R. Meech P. Laxminarayan B. Kitts, B. LeBlanc. Click Fraud. *Bulletin of the American Society for Information Science and Technology*, 32:14–16, 2005.

[18] M. Backes, B. Pfitzmann, and M. Schunter. A Toolkit for Managing Enterprise Privacy Policies. In *Proceedings of 8th European Symposium on Research in Computer Security (ESORICS)*, volume 2808 of *Lecture Notes in Computer Science*, pages 162–180. Springer, October 2003.

[19] M. Barbaro. A Face Is Exposed for AOL Searcher No. 4417749. The New York Times, available online at http://wa.la/3E1, August 2006.

[20] M. Bartiromo. Inside Google. CNBC, available online at http://wa.la/3E0, December 2009.

[21] S. M. Beitzel, E. C. Jensen, A. Chowdhury, D. Grossman, and O. Frieder. Hourly analysis of a very large topically categorized web

query log. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 321–328, New York, NY, USA, 2004. ACM.

[22] O. Berthold, H. Federrath, and S. Köpsell. Web MIXes: A system for anonymous and unobservable Internet access. In H. Federrath, editor, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, pages 115–129. Springer-Verlag, LNCS 2009, July 2000.

[23] N. Borisov, I. Goldberg, and E. Brewer. Off-the-record communication, or, why not to use PGP. In *WPES '04: Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, pages 77–84, New York, NY, USA, 2004. ACM Press.

[24] M. Boyle. A Shared Vocabulary for Privacy. In *Workshop on Ubicomp Communities: Privacy as Boundary Negotiation. Held as part of the UBICOM'2003 5th International Conference on Ubiquitous Computing*, October 2003.

[25] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

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

[27] D. Chaum. The dining cryptographers problem: unconditional sender and recipient untraceability. *J. Cryptol.*, 1(1):65–75, 1988.

[28] A. Cooper. A survey of query log privacy-enhancing techniques from a policy perspective. *ACM Trans. Web*, 2(4):1–27, 2008.

[29] K. Coyle. P3P: Pretty Poor Privacy? A Social analysis of the Platform for Privacy Preferences, June 1999 1999.

[30] L. Cranor. P3P: Making Privacy Policies More Useful. *IEEE Security and Privacy*, 01(6):50–55, 2003.

[31] L. Cranor, M. Langheinrich, M. Marchiori, and J. Reagle. The Platform for Privacy Preferences 1.1 (P3P1.1) Specification. W3C Recommendation, 2005.

[32] L. Cranor and J. Reagle. The Platform for Privacy Preferences. *Communications of the ACM*, 4(2):48 – 55, February 1999. Also in Japanese at IPSJ (Information Processing Society of Japan) Magazine. Vol.40 No.7 July 1999; also as W3C NOTE. 31 July 1998.

[33] G. Danezis, R. Dingledine, D. Hopwood, and N. Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *In Proceedings of the 2003 IEEE Symposium on Security and Privacy*, pages 2–15, 2003.

[34] N. Daswani and M. Stoppelman. The anatomy of Clickbot.A. In *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, pages 11–11, Berkeley, CA, USA, 2007. USENIX Association.

[35] J. D. Day and H. Zimmerman. The OSI Reference Model. *Proceedings of the IEEE*, 71:1334–1340, 1983.

[36] J. DeCew. Privacy. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Fall 2006.

[37] P. Deutsch and A. Emtage. The archie System: An Internet Electronic Directory Service. *Connexions*, 6(2), 1992.

[38] M. Dewey. *Decimal Classification & Relative Index*. Forest Press, 1919.

[39] C. Díaz, S. Seys, J. Claessens, and B. Preneel. Towards measuring anonymity. In R. Dingledine and P. Syverson, editor, *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.

[40] T. Dierks and C. Allen. The TLS Protocol Version 1.0. RFC 2246 (Proposed Standard), January 1999. Obsoleted by RFC 4346, updated by RFC 3546.

[41] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.1. RFC 4346 (Proposed Standard), April 2006. Obsoleted by RFC 5246, updated by RFCs 4366, 4680, 4681.

[42] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[43] B. Edelman, M. Ostrovsky, and M. Schwarz. Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords. *The American Economic Review*, 97(1):242–259, March 2007.

[44] D. Eroshenko. Click fraud. The state of the industry. *Pay per Click Analyst*, pages 10 – 19, 2004.

[45] D. Fahrenholtz and W. Lamesdorf. Transactional Security for a Distributed Reputation Management System. In *Proceedings of the Third International Conference on E-Commerce and Web Technologies (EC-Web)*, volume LNCS 2455, pages 214 – 223. Springer, September 2002.

[46] C. Faloutsos. Access Methods for Text. *ACM Comput. Surv.*, 17(1):49–74, 1985.

[47] H. Federrath. Privacy Enhanced Technologies: Methods - Markets - Misuse. In *TrustBus*, pages 1–9, 2005.

[48] W. Feller. *An Introduction to Probability Theory and Its Applications, Volume 1.* Wiley, January 1968.

[49] E. Felten and M. Schneider. Timing attacks on Web privacy. In *ACM Conference on Computer and Communications Security*, pages 25–32, 2000.

[50] M. J. Freedman and R. Morris. Tarzan: A Peer-to-Peer Anonymizing Network Layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, Washington, D.C., November 2002.

[51] A. O. Freier, P. Karlton, and P. C. Kocher. *SSL Version 3.0*, 12 1995. Internet Draft.

[52] E. Gabber, P. B. Gibbons, D. M. Kristol, Y. Matias, and A. Mayer. On secure and pseudonymous client-relationships with multiple servers. *ACM Trans. Inf. Syst. Secur.*, 2(4):390–415, 1999.

[53] E. Gabber, P. B. Gibbons, Y. Matias, and A. J. Mayer. How to Make Personalized Web Browising Simple, Secure, and Anonymous. In *Financial Cryptography*, pages 17–32, 1997.

[54] L. Garcés-Erice, K. W. Ross, E. W. Biersack, P. Felber, and G. Urvoy-Keller. Topology-Centric Look-Up Service. In *Networked Group Communication*, pages 58–69, 2003.

[55] R. Gavison. Privacy and the limits of the law. In D. G. Johnson and H. Nissenbaum, editor, *Computers, Ethics, and Social Values*, pages 332–351. Prentice Hall, 1995.

[56] I. Goldberg. Privacy-enhancing Technologies for the Internet, II: Five Years Later. In R. Dingledine and P. Syverson, editors, *Proceedings of Privacy Enhancing Technologies workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.

[57] I. Goldberg. Privacy-enhancing Technologies for the Internet III: Ten Years Later. In A. Acquisti, S. Gritzalis, C. Lambrinoudakis, and S. di Vimercati, editors, *Digital Privacy: Theory, Technologies, and Practices*. Auerbach Publications (Taylor and Francis Group), 2007.

[58] I. Goldberg, D. Wagner, and E. Brewer. Privacy-enhancing Technologies for the Internet. In *Proc. of 42nd IEEE Spring COMPCON*. IEEE Computer Society Press, February 1997.

[59] D. Goldschlag, M. Reed, and P. Syverson. Onion Routing for Anonymous and Private Internet Connections. *Communications of the ACM*, 42:39–41, 1999.

[60] M. Götz, A. Machanavajjhala, H. Wang, X. Xiao, and J. Gehrke. Privacy in Search Logs. *CoRR*, abs/0904.0682, 2009.

[61] C. G. Günther. An identity-based key-exchange protocol. In *EUROCRYPT '89: Proceedings of the workshop on the theory and application of cryptographic techniques on Advances in cryptology*, pages 29–37, New York, NY, USA, 1990. Springer-Verlag New York, Inc.

[62] Q. Guo and E. Agichtein. Exploring mouse movements for inferring query intent. In *SIGIR*, pages 707–708, 2008.

[63] M. Gupta, P. Judge, and M. Ammar. A reputation system for peer-to-peer networks. In *Proceedings of the 13th international workshop on Network and operating Systems support for digital audio and video (NOSSDAV)*, 2003.

[64] L. Hall. Inventory of privacy-enhancing technologies (PETs). Report DSTI/ICCP/REG(2001)1/FINAL, Working Party on Information Security and Privacy, Organisation for Economic Co-operation and Development, 2002.

[65] H. Hochheiser. The Platform for Privacy Preference as a Social Protocol: An Examination Within the US Policy Context. *ACM Transactions on Internet Technology (TOIT)*, 2(4):276 – 306, November 2002.

[66] D. Howe and H. Nissenbaum. TrackMeNot: Resisting Surveillance in Web Search. In *Lessons from the Identity Trail: Anonymity, Privacy, and Identity in a Networked Society*, pages 417–436. Oxford University Press, 2009.

[67] L. Introna and A. Pouloudi. Privacy in the Information Age: Stakeholders, Interests and Values. *Journal of Business Ethics*, 22:27 – 38, 1999.

[68] J. Cho and H. Garcia-Molina. The Evolution of the Web and Implications for an Incremental Crawler. Technical Report 1999-22, Stanford InfoLab, 1999.

[69] M. Jakobsson. Flash Mixing. In *In Principles of Distributed Computing - PODC '99. ACM*, pages 83–89. ACM, 1999.

[70] B. J. Jansen. Adversarial information retrieval aspects of sponsored search. *Adversarial Information Retrieval on the Web Workshop at SIGIR 2006*, 2006.

[71] B. J. Jansen and A. Spink. How are we searching the World Wide Web? A comparison of nine search engine transaction logs. *Inf. Process. Manage.*, 42(1):248–263, 2006.

[72] N. F. Johnson and S. Jajodia. Exploring Steganography: Seeing the Unseen. *IEEE Computer*, pages 26–34, February 1998.

[73] A. Jøsang, R. Ismail, and C. Boyd. A Survey of Trust and Reputation Systems for Online Service Provision. *Decision Support Systems*, 43(2):618 – 644, March 2007.

[74] D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and R. Panigrahy. Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web. In *ACM Symposium on Theory of Computing*, pages 654–663, May 1997.

[75] G. Karjoth, M. Schunter, and M. Waidner. Platform for Enterprise Privacy Practices: Privacy-Enabled Management of Customer Data. In *Privacy Enhancing Technologies*, pages 69–84, 2002.

[76] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing Search Queries and Clicks Privately. In *18th International World Wide Web Conference (WWW2009)*, April 2009.

[77] M. Koster. ALIWEB - Archie-like Indexing in the WEB. *Computer Networks and ISDN Systems*, 27(2):175–182, 1994.

[78] M. Langheinrich, L. Cranor, and M. Marchiori. APPEL: A P3P Preference Exchange Language. W3C Working Draft, April 2002.

[79] M. Leino-Kilpi. Privacy: a review of the literature. *International journal of nursing studies*, 38(6):663–71, 2002. Affiliaton: University of Turku, Finland. heleno.leino-kilpi@utu.fi.

[80] B. Levine and C. Shields. Hordes: A Multicast Based Protocol for Anonymity. *Journal of Computer Security*, 10(3):213–240, 2002.

[81] B. N. Levine, M. Reiter, C. Wang, and M. Wright. Timing Attacks in Low-Latency Mix Systems. In *Proc. Financial Cryptography (FC) (LNCS 3110)*, pages 251—265, February 2004.

[82] L. Macaulay. Privacy Enhancing Technologies - State of the Art Review Version 1. Technical Report TRS-2002-001, Computation Department, University of Manchester Institute of Science and Technology, Manchester, UK, 2002.

[83] E. P. Markatos. On caching search engine query results. *Computer Communications*, 24(2):137–143, 2001.

[84] C.P. McParland and R. Connolly. Empirical Research on Technology-Related Privacy Concerns: A Review and Critical Assessment. In *16th European Conference on Information Systems (Golden W, Acton T, Conboy K, van der Heijden H, Tuunainen VK eds.)*, pages 658 – 668, Galway, Ireland, 2008.

[85] E. Nakashima. AOL Search Queries Open Window Onto Users' Worlds. Washington Post, available online at http://wa.la/3DZ, August 2006.

[86] G. Ng-Kruelle, J. Felix Hampe, P. A. Swatman, and D. S. Rebne. The Price of Convenience: Privacy and Mobile Commerce. *Quarterly Journal of Electronic Commerce*, 3(3):273–385, 2002.

[87] O. Berthold and H. Federrath and M. Köhntopp. Project "anonymity and unobservability in the Internet". In *CFP '00: Proceedings of the*

tenth conference on Computers, freedom and privacy, pages 57–65, New York, NY, USA, 2000. ACM.

[88] R. Olbrich and C. D. Schultz. Search Engine Marketing and Click Fraud, 2008.

[89] M. S. Olivier. A Layered Architecture for Privacy-enhancing Technologies. In Jan H P Eloff, Hein S Venter, Les Labuschagne, and Mariki M Eloff, editors, *Proceedings of the Third Annual Information Security South Africa Conference (ISSA2003)*, pages 113–126, Sandton, South Africa, July 2003.

[90] M. S. Olivier. Flocks: Distributed Proxies for Browsing Privacy. In Gary Marsden, Paula Kotzé, and Ayodele Adesina-Ojo, editors, *Proceedings of SAICSIT 2004 — fulfilling the promise of ICT*, pages 79–88, Stellenbosch, South Africa, October 2004.

[91] M. S. Olivier. Privacy under Conditions of Concurrent Interaction with Multiple Parties. In de Capitani di Vimercati, Sabrina and Indrakshi Ray and Indrajit Ray, editor, *Data and Applications Security XVII — Status and Prospects*, pages 105–118. Kluwer, 2004.

[92] L. Øverlier and P. Syverson. Locating Hidden Servers. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 100–114. IEEE Computer Society, May 2006.

[93] G. Pass, A. Chowdhury, and C. Torgeson. A picture of search. In *Infoscale*, page 1, 2006.

[94] A. Pfitzmann and M. Koehntopp. Anonymity, unobservability, and pseudonymity – a proposal for terminology. In *Proceedings of the International Workshop on Design Issues in Anonymity and Unobservability*, volume 2009/2001 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2001.

[95] A. Pfitzmann and M. Waidner. Networks Without User Observability. *Computers and Security*, 2(6):158–166, 1987.

[96] B. Pinkerton. Finding What People Want: Experiences with the WebCrawler. In Anonymous, editor, *Proceedings of the 2nd International World Wide Web*, volume 18(6) of *Online & CDROM review: the international journal of*, Medford, NJ, USA, 1994. Learned Information.

[97] J. Postel and J. Reynolds. File Transfer Protocol. RFC 959 (Standard), October 1985. Updated by RFCs 2228, 2640, 2773, 3659.

[98] J. Postel and J.K. Reynolds. Telnet Protocol Specification. RFC 854 (Standard), May 1983. Updated by RFC 5198.

[99] M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Anonymous Connections and Onion Routing. *IEEE Journal on Selected Areas in Communications*, 16(4):482–494, May 1998.

[100] M. K. Reiter and A. D. Rubin. Crowds: anonymity for Web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[101] P. Resnick and R. Zeckhauser. Trust Among Strangers in Electronic Transactions: Empirical Analysis of eBay's Reputation System. *Advances in Applied Microeconomics*, 11, 2002.

[102] P. Resnick, R. Zeckhauser, J. Swanson, and K. Lockwood. The Value of Reputation on eBay: A Controlled Experiment, 2003.

[103] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 26(1):96–99, 1983.

[104] M. Rotenberg. Fair information practices and the architecture of privacy. *Stanford Technology Law Review*, 1, 2001.

[105] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, November 2001.

[106] A. Rutherford, R. A Botha, and M. S Olivier. Towards a Hippocratic Log File Architecture. In Gary Marsden and Paula Kotzé and Ayodele Adesina-Ojo, editor, *Proceedings of SAICSIT 2004 — fulfilling the promise of ICT*, pages 186–193, Stellenbosch, South Africa, October 2004.

[107] A. Rutherford, R. A Botha, and M. S Olivier. Towards Hippocratic Log files. In *Proceedings of the Fourth Annual Information Security South Africa Conference (ISSA2004)*, Midrand, South Africa, June/July 2004. Published electronically.

[108] S. Garfinkel. *PGP: Pretty Good Privacy.* O'Reilly & Associates, Inc., Sebastopol, CA, USA, 1996.

[109] V. Seničar, B. Jerman-Blažič, and T. Klobučar. Privacy-enhancing technologies: approaches and development. *Comput. Stand. Interfaces*, 25(2):147–158, 2003.

[110] C. Shields. Responder Anonymity and Anonymous Peer-to-Peer File Sharing. In *ICNP '01: Proceedings of the Ninth International Conference on Network Protocols*, page 272, Washington, DC, USA, 2001. IEEE Computer Society.

[111] D. J. Solove. Conceptualizing Privacy. *California Law Review*, 90:1087, 2002.

[112] D. J. Solove. "I've Got Nothing to Hide" and Other Misunderstandings of Privacy. *San Diego Law Review*, 44:745, 2007.

[113] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proceedings of the ACM SIGCOMM '01 Conference*, San Diego, California, August 2001.

[114] R. Thibadeau. A Critique of P3P: Privacy on the Web, August 2000.

[115] J. J. Thomson. The Right to Privacy. *Philosophy and Public Affairs*, 4(4):295–314, 1975.

[116] V. Bellotti and A. Sellen. Design for Privacy in Ubiquitous Computing Environments. In *Proceedings of the Third European Conference on Computer Supported Cooperative Work (ECSCW'93)*, pages 77–92. Kluwer, 1993.

[117] R. G. van Schyndel, A. Z. Tirkel, and C. F. Osborne. A Digital Watermark. In *ICIP (2)*, pages 86–90, 1994.

[118] D. Wagner and B. Schneier. Analysis of the SSL 3.0 protocol. In *In Proceedings of the Second UNIX Workshop on Electronic Commerce*, pages 29–40. USENIX Association, 1996.

[119] S. Warren and L. Brandeis. The Right to Privacy. *Harward Law Review*, 4:193–220, 1890.

[120] A. F. Westin. *Privacy and Freedom.* Atheneum, New York, 1967.

[121] C. K. Wilbur and Y. Zhu. Click Fraud. *Marketing Science*, 28(2):293–308, 2009.

[122] J. Wright, S. Stepney, J. A. Clark, and J. L. Jacob. Designing Anonymity - A Formal Basis for Identity Hiding. Internal Yellow Report, York University, York, UK, December 2004.

[123] M. Wright, M. Adler, B. N. Levine, and C. Shields. Defending Anonymous Communication Against Passive Logging Attacks. In *Proc. IEEE Symposium on Security and Privacy (Oakland)*, pages 28–41, May 2003.

[124] M. K. Wright, M. Adler, B. N. Levine, and C. Shields. An Analysis of the Degradation of Anonymous Protocols. In *Proc. ISOC Symposium Network and Distributed System Security (NDSS)*, pages 38–50, February 2002.

[125] M. K. Wright, M. Adler, B. N. Levine, and C. Shields. The Predecessor Attack: An analysis of a threat to anonymous communications systems. *ACM Trans. Inf. Syst. Secur.*, 7(4):489–522, 2004.

[126] G. Zacharia and P. Maes. Trust management through reputation mechanisms. *Applied Artificial Intelligence*, 14(19):881 – 907, 2000.

[127] B. Zhao, J. Kubiatowicz, and A. D. Joseph. Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing. Technical Report UCB/CSD-01-1141, University of California Berkeley, Electrical Engineering and Computer Science Department, April 2001.

[128] Y. Zhu, L. Xiong, and C. Verdery. Anonymizing user profiles for personalized web search. In *WWW*, pages 1225–1226, 2010.