Flocks: Distributed Proxies for Browsing Privacy

MARTIN S OLIVIER
University of Pretoria

This paper introduces a Privacy-Enhancing Technology (PET) based on a hybrid of Crowds and anonymising proxies. The PET — referred to as Flocks — operates by establishing a number of Web proxies and letting these proxies randomly forward requests to other proxies (or the final destination). This distributes users' requests over a number of such proxies, thereby helping to protect their (browsing) privacy.

The problem that the paper considers is the effect of two primary design parameters on the privacy of the overall system. These parameters are the probability with which a proxy sends a request to the destination server rather than another proxy (α) and the number of proxies in the system (N). Two privacy objectives are identified, namely the number of hops used to satisfy a request and the portion of proxies that 'know' about a request. A third requirement deals with the external communication costs of the system. A formal analysis is performed to determine these three factors from the two identified parameters. Finally numerical examples are used to explore the impact of these two parameter choices in concrete terms.

The proposed PET differs from existing PETs in two significant manners: It is primarily intended to be used inside an organisation to protect the privacy of users inside the organisation (in particular, employees) and it takes explicit cognisance of forensic factors.

Categories and Subject Descriptors: H.3.5 [**Online Information Services**]: Data sharing—*Security*; H.2.7 [**Database Administration**]: Security, integrity, and protection; K.4.1 [**Public Policy Issues**]: Privacy

General Terms: Management, Reliability, Security

Additional Key Words and Phrases: Personal privacy, privacy architecture, privacy-enhancing technologies

1. INTRODUCTION

The fact that one's every action is logged when using the Internet has been a concern for many years [Cranor 1999]. Such logging implies that a user can be profiled. In addition to profiling a user's general behaviour, a single visit to a single dubious Web site can be easily associated with a user in perpetuity.

A range of Privacy-enhancing Technologies (PETs) have been introduced over the years to address the threat posed by such logging of a user's actions. These include anonymising proxies, such as Anonymizer [Caloyannides 2000], LPWA [Gabber et al. 1999] and Janus [Rieke and Demuth 2001]. These proxies will accept a Web access request from a user and then submit the request to the actual server. The proxy's details will then be logged at the server, rather than the user's details. (Note that many of these proxies also provide additional mechanisms to protect the user's privacy.) Even a typical proxy used by many organisations at the gateway between the organisation's employees and the Internet will hide (some of) the user's details from the target server, and will therefore help to protect the privacy of such a user.

A major concern that is, however, often levied against such proxies is the fact that the proxy is now able to log the user's requests. This could, in principle, be more dangerous than the original situation, because the proxy will be able to log *all* the user's private Web requests, whereas surfing without such a proxy would have logged these requests at the various servers. If all private information is concentrated at a single point, the information becomes more valuable and therefore a more likely point of attack.

Another form of PET that has been proposed to solve problems with such logging operates on the network layer. When a request is sent, it is sent to the first of a number of anonymising routers. Each router either forwards the request to the final destination or to the next such router. When the request is forwarded, it is rewritten as if the request came from that router. Hence requests will be logged at the destination server, using the details of the last router en route (and this could vary over time). Moreover, the various routers do not know whether the router or host that each received the request from was the machine at which the request originated, or another router that merely relayed the request. Therefore logging at the various routers does not disclose any significant information about the actions of the individual users. The decision about whether to forward the request to the final destination or another router can be made on a random or predetermined basis. In Crowds [Reiter and Rubin 1999] the decision is made randomly. In Onion routing [Goldschlag et al. 1999] the sender

Author Address:
M.S. Olivier, Information and Computer Security Architectures (ICSA) Research Group, Department of Computer Science, University of Pretoria, Pretoria, 0002, South Africa; http://mo.co.za.

of a message predetermines the route it will follow and uses encryption so that each router only knows the next destination to send the message to.

Hiding users' details at the network layer — as these anonymising routers do — presents other problems. Firstly, it becomes less feasible to use a local Web proxy in an organisation for reasons of efficiency. This will be considered in more detail later in the paper. Secondly, it is known that techniques such as Onion routing make it particularly hard to perform a forensic analysis — where a legitimate reason exists for tracing the sender of a message [Forte 2002].

Given the problems associated with anonymising proxies and anonymising routers, this paper will consider a fairly straightforward hybrid of the two techniques discussed above: We will use a number of caching proxies that will each forward the request to either another such proxy, or to the final destination. A random decision will be made at each proxy to decide which of the two actions to perform for each request. We will refer to this model as *Flocks*. Based on the analysis to be performed below, Flocks is primarily intended to be used inside an organisation. The intention is to protect the individual from prying eyes inside the organisation (and not the fact that someone from the organisation visited a particular site). The name *Flocks* is intended to reflect anonymity within the group, where the group might be identifiable. In addition to the level of privacy the system provides, the cost associated with this privacy will also be considered. Cost will be measured in terms of additional requests that have to be sent to servers outside the organisation due to the use of this system.

There are clearly two major parameters that will influence the effectiveness of the system: (1) the number of proxies, and (2) the probability that a request will be sent to the final destination (and not forwarded to another proxy). Let N represent the former and α represent the latter.

For purposes of privacy, it is clear that N should be as large as possible: The more proxies there are in the system, the less likely it is that enough of them will be compromised, their logs aggregated and a composite picture of user behaviour created. Costs, on the other hand, imply that N should be relatively small.

It would initially seem that a small value for α would be best. A small α will imply that the request would, on average, be forwarded many times before it is sent to the destination server. It seems as if this would make it relatively hard to track the request to its origin. This, however, is only part of the issue. It is also true that the longer such a trail becomes, the more 'footprints' it will leave in various proxies and the greater the probability that someone will notice the request. It should also be noted that the length of the trail holds cost implications: The more often a request is forwarded, the more resources are consumed and the longer the request may take to be responded to. However, for the purposes of this paper, we will assume that costs for communication between proxies are negligible, as is the time taken to communicate between proxies. This would be the case where the proxies exist on a local area network inside an organisation, and where the internal network is fast in comparison to the wide area network that connects it to the outside world. Hence, we are interested in a value for α that is determined by privacy considerations, rather than internal communication costs.

The purpose of this paper is to explore the ideal values for N and α.

Clearly, the optimum values for N and α will be influenced by a range of factors — including the number of requests to be handled by the network, the frequency with which requests are repeated by different users and (external) communication costs. The paper will therefore not attempt to determine optimum values for these two parameters, but to determine how such values could be calculated.

The remainder of this paper is structured as follows. Section 2 will specify the properties to be used as criteria and derive the formulae to calculate those properties. Section 3 will use numerical examples to discuss the implications of the derived formulae in concrete terms. Section 3 will also discuss a number of other issues related to the proposed model. Section 4 will conclude the paper.

2. ANALYSIS

We begin this section by introducing terminology that will be used below.

The term *trail* has already been introduced to refer to the nodes (proxies) on the path from the first proxy where a request is submitted to the last (before the request is sent to the external server). Let ν indicate the average length of a trail. ν will be calculated below.

Let *saturation* refer to the proportion of proxy nodes that have a specific page cached. Over time, requests for pages will find the page they refer to cached at a node that forms part of the request's trail. Let $\sigma_{n,p}$ represent the average proportion of proxy servers that have a specific page p cached after n requests for that page. Since the page p will be implied in what follows below, we will omit the p when talking about saturation, and only write σ_n with the same meaning as $\sigma_{n,p}$. At some point the probability of finding such a page p cached will approach 1. At this point, it will not be necessary to send further requests to the external server. Let Σ_ε be the lowest saturation for which this probability exceeds $1 - \varepsilon$. Since ε will also be implied in what follows, we will generally only write Σ.

The analysis that follows will be based on the following observations:

(1) The average trail length ν should be long enough to involve a sufficient number of proxy servers. If too few proxy

servers are involved in a typical request, it will be simple to associate a sender of a message with a log entry and privacy will not be sufficiently protected.

(2) ν/N should be small. If this ratio is too large, a request will pass through too many proxies, where the fact that such a request has been issued may be noticed. This is particularly applicable when a user issues an unusual request; such unusual requests — ie requests that are not issued by the 'average' user — are bound to be the most privacy-sensitive ones.

(3) For reasons of cost, the saturation point Σ should be reached as soon as possible; ie the smallest value n for which $\sigma_n \geq \Sigma$, should be as small as possible.

Below we will refer to these observations as the privacy properties. While these principles have been selected based on the balance between cost and privacy, they will be useful in the analysis below, and not only as criteria to evaluate specific examples.

2.1 Average trail length

In this section we calculate the average length of a trail. Let τ_n denote a trail of length n and let $P(\tau_n)$ indicate the probability of a trail of length n. Then the average (or mean) length of a trail, ν will be

$$\sum_{i=1}^{\infty} i \cdot P(\tau_i) \tag{1}$$

Clearly a trail of length 1 will only occur if the first proxy contacted forwards the request to the external server. According to our assumption any proxy (and therefore also the first) will do this with probability α. Hence $P(\tau_1) = \alpha$.

A trail of length 2 will only occur if the first proxy does not send the request to the external server, but the second does. This will happen with probability $1-\alpha$ at the first server and probability α at the second server. Hence $P(\tau_2) = (1-\alpha)\cdot\alpha$.

Extending this reasoning, it is clear that, in general, $P(\tau_n) = (1-\alpha)^{n-1} \cdot \alpha$ for $n \geq 1$.

From equation 1 we get

$$\begin{aligned}
\nu &= \sum_{i=0}^{\infty} i \cdot (1-\alpha)^{i-1} \cdot \alpha \\
&= \frac{\alpha}{1-\alpha} \sum_{i=0}^{\infty} i \cdot (1-\alpha)^{i} \\
&= \frac{\alpha}{1-\alpha} \cdot \frac{1-\alpha}{(1-(1-\alpha))^2}
\end{aligned}$$

The previous step follows from the formula for series expansion of this form (see equation 12 in the appendix). When simplified, this yields

$$\nu = \frac{1}{\alpha} \tag{2}$$

In many situations it will also be useful to know the median trail length, which we will refer to as μ.

Let τ_n^* indicate all trails of length n or less. Then

$$\begin{aligned}
P(\tau_n^*) &= \sum_{i=1}^{n} P(\tau_i) \\
&= \sum_{i=1}^{n} (1-\alpha)^{i-1}\alpha \\
&= \alpha \frac{1-(1-\alpha)^n}{1-(1-\alpha)} \\
&= 1-(1-\alpha)^n
\end{aligned}$$

We can now calculate the average trail length as the point where the probability of a longer or a shorter trail are the same. Then $P(\tau_\mu^*) = \frac{1}{2}$. From this follows that

$$\mu = -\log_{1-\alpha} 2 \tag{3}$$

It can be shown that $\mu < \nu$ for all values of $\alpha \in (0,1)$. This means that the typical trail will be shorter than the average trail. This is due to the fact that extremely long trails, although improbable, are possible. Such infrequent long trails will influence the mean, but not the median in any significant way.

2.2 The required saturation

Remember that saturation refers to the ratio of proxies that have cached the page under consideration. We want to determine the saturation threshold $0 \leq \Sigma \leq 1$ for some small ε such that the probability for satisfying the request from a cached copy exceeds $1 - \varepsilon$.

The situation that the request cannot be satisfied from the cache will only occur if the request is forwarded to a proxy that does not contain the page for each hop in the trail. Given the fact that the trail has an average length of ν, the probability of this occurring is $(1 - \Sigma)^\nu$. Therefore, the probability that the request will indeed be satisfied from the cache is

$$1 - (1 - \Sigma)^\nu$$

The threshold at which this is close enough to 1 occurs when

$$1 - (1 - \Sigma)^\nu = 1 - \varepsilon$$

Therefore

$$\Sigma = 1 - \varepsilon^{\frac{1}{\nu}} \tag{4}$$

While this expression has been derived in terms of ν, μ could also have been used. In the analysis below, calculations will indeed be based on μ.

2.3 Expected saturation

As noted above σ_n, with $0 \leq \sigma_n \leq 1$, is the expected saturation after request n for the page under consideration has been completed. Our intention in this section is to derive a formula for σ_n.

Assume the saturation after round $n - 1$ is σ_{n-1}. We want to determine the probability that the saturation is increased by k/N for some k after request n has been served. More formally, we want to determine $P(\sigma_n = \sigma_{n-1} + \frac{k}{N})$. For the sake of simplicity, we will denote this probability by writing $P(\sigma, n, +k)$.

Clearly $P(\sigma, n, +0) = \sigma_{n-1}$, since the probability that the page is found cached at a first proxy is equal to the proportion of proxies that already have the page cached, ie σ_{n-1}. $P(\sigma, n, +1)$ depends on the probability that the first proxy contacted does not have the page cached, but that this proxy then either forwards the request to the external server (with probability α), or chooses to forward the request to another proxy (with probability $1 - \alpha$) *and* chooses a proxy that has the page cached (with probability σ_{n-1}). Therefore

$$P(\sigma, n, +1) = (1 - \sigma_{n-1})[\alpha + (1 - \alpha)(\sigma_{n-1})]$$

Similarly, $P(\sigma, n, +2)$ depends on choosing a node without the requested page (probability $1 - \sigma_{n-1}$), choosing to refer it to another proxy node (probability $1 - \alpha$) *and* then choosing another node without the page cached (probability $1 - \sigma_{n-1}$), after which the choice between the external server and a local cached copy has to be made, as was the case for $P(\sigma, n, +1)$. Therefore

$$P(\sigma, n, +2) = (1 - \sigma_{n-1})(1 - \alpha)(1 - \sigma_{n-1})[\alpha + (1 - \alpha)(\sigma_{n-1})]$$

In general,

$$P(\sigma, n, +k) = (1 - \sigma_{n-1})^k(1 - \alpha)^{k-1}[\alpha + (1 - \alpha)(\sigma_{n-1})]$$

The expected number of additional nodes that will contain the page after request n has been processed is k, where

$$\begin{aligned}
k &= \sum_{i=0}^{\infty} i \cdot P(\sigma, n, +i) \\
&= \sum_{i=0}^{\infty} i \cdot (1 - \sigma_{n-1})^k(1 - \alpha)^{k-1}[\alpha + (1 - \alpha)(\sigma_{n-1})] \\
&= \sum_{i=0}^{\infty} i \cdot [(1 - \sigma_{n-1})(1 - \alpha)]^k[\frac{\alpha}{1 - \alpha} + (1 - \alpha)(\sigma_{n-1})] \\
&= (\frac{\alpha}{+}(1 - \alpha)\sigma_{n-1}) \cdot \frac{(1 - \sigma_{n-1})(1 - \alpha)}{[1 - (1 - \sigma_{n-1})(1 - \alpha)]^2} \\
&= \frac{\alpha + (1 - \alpha)\sigma_{n-1}}{1 - \alpha} \cdot \frac{(1 - \sigma_{n-1})(1 - \alpha)}{[1 - (1 - \sigma_{n-1})(1 - \alpha)]^2]} \\
&= \frac{[\alpha + (1 - \alpha)\sigma_{n-1}](1 - \sigma_{n-1})}{[1 - (1 - \sigma_{n-1})(1 - \alpha)]^2]}
\end{aligned}$$

Let $\overline{\alpha}$ denote $1-\alpha$ and $\overline{\sigma}_{n-1}$ denote $1-\sigma_{n-1}$. Then

$$k=\frac{(\alpha+\overline{\alpha}\sigma_{n-1})\overline{\sigma}_{n-1}}{(1-\overline{\sigma}_{n-1}\overline{\alpha})^2} \qquad (5)$$

Therefore

$$\begin{aligned}\sigma_n &= \sigma_{n-1}+\frac{k}{N}\\ &= \sigma_{n-1}+\frac{(\alpha+\overline{\alpha}\sigma_{n-1})\overline{\sigma}_{n-1}}{(1-\overline{\sigma}_{n-1}\overline{\alpha})^2N}\end{aligned} \qquad (6)$$

To calculate any value for σ_n it is necessary to determine the value of σ_1. Nothing prevents one defining $\sigma_0=0$ and using equation 6 to calculate σ_1. For this case

$$\begin{aligned}k &= \frac{(\alpha+\overline{\alpha}\sigma_{n-1})\overline{\sigma}_{n-1}}{(1-\overline{\sigma}_{n-1}\overline{\alpha})^2}\\ &= \frac{(\alpha+\overline{\alpha}\cdot 0)1}{(1-1\overline{\alpha})^2}\\ &= \frac{\alpha}{\alpha^2}\\ &= \nu\end{aligned}$$

As was the case when the average trail length (ν) was considered above, it should be pointed out that this average value for σ_1 is influenced by the very few extremely long trails that can occur. We will therefore use the typical case (rather than the mean case) for an estimate for σ_1.

If μ is sufficiently small (compared to N), the initial (median) μ nodes selected in the first trail will not contain any duplicates and will result in a saturation of μ/N. More formally, if $N\gg\mu$, then

$$\sigma_1=\mu/N \qquad (7)$$

The conditions for $N\gg\mu$ will be considered later. This is the 'typical' case in the sense that it is based on median values and most values will cluster around the median.

2.4 On $N\gg\mu$ for σ_1

In the previous section it was stated that if $N\gg\mu$, then μ/N is a good estimate for σ_1 (equation 7). Here we consider what $N\gg\mu$ implies for the estimate to be valid.

It is clear that μ will be a good estimate for σ_1 only if the trail of the first request never revisits any proxy. Therefore this trail has to be a permutation of some of the set of available proxies. It is safe to use μ as an estimate, if the probability of such a permutation is larger than the combined probability of all other sequences in that trail. This implies that this probability should be larger than 0.5. The number of permutations of length μ from the set of N proxies is $N!/(N-\mu)!$. We will write this number as $N^{!\mu}$. The number of all possible sequences from the set of N proxies is N^{μ}. Therefore, our estimate of σ_1 is valid if

$$\frac{N^{!\mu}}{N^{\mu}}\geq 0.5 \qquad (8)$$

Note that the restriction placed by equation 8 is not a necessary restriction for Flocks to be used; it is only a restriction that validates the estimate of σ_1 as μ. The analysis could also have been done with a more conservative estimate of σ_1. The current estimate is, however, sufficient for the purposes of this paper — in particular given our assumption (or observation) 2 in Section 2 that implies $N\gg\nu>\mu$.

3. DISCUSSION

This section will consider the implications of some of the formulae derived above by considering some concrete possible values for the various parameters. After this, some remaining issues will be discussed.

3.1 Some concrete values

The values for the average trail lengths ν and μ for some values of α are given in Table I. As noted in the introduction of this paper, specific choices will depend on specific circumstances. Under typical circumstances there does not seem to be a need for an excessively long average trail length. It has already been argued above that a too long average trail poses its own risks. From Table I it seems that values for α in the range $[0.05, 0.1]$ would be most useful, yielding median trail lengths in the range $[6.5, 13.5]$. Trails in this range would, typically, require collusion of between 6/2 and 14/2 (specific) custodians of proxies to trace a request to a specific user. While collusion on such a scale is improbable, this is within

α	μ	ν
0.01	68.9	100.0
0.02	34.3	50.0
0.03	22.7	33.3
0.05	13.5	20.0
0.07	9.5	14.2
0.08	8.3	12.5
0.10	6.5	10.0
0.20	3.1	5.0
0.30	1.9	3.3
0.50	1.0	2.0

Table I. Average trail lengths for some values of α.

μ	**N**
3	6
5	17
7	33
8	43
10	70
13	117
14	136
23	373
34	821

Table II. Threshold values for N such that $N \gg \mu$.

ε	$\mathbf{\Sigma}$	
	$\mu = \mathbf{13.5}$	$\mu = \mathbf{6.5}$
0.01	0.288	0.503
0.001	0.400	0.650
0.0001	0.494	0.753
0.00001	0.573	0.826
0.000001	0.640	0.877

Table III. Values of Σ for some values of ε

easy reach of a (legitimate) forensic examination. Below we will most often use $\alpha = 0.05$, with a median trail length of $\nu = 13.5$ as an example.

Table II lists the values of N for various values of μ as required for $N \gg \mu$ according to equation 8 so that our analysis is valid. Note that N grows rapidly and for a somewhat larger μ an alternative analysis — in particular, a different estimate for σ_1 — might be warranted. Reasons have, however, already been given for preferring an average trail length that is not too long.

For the example case under consideration, $\mu = 13.5$. For this value the threshold value of N is between 117 and 136 indicating that the number of proxies should be in this region or higher. At $N = 136$, the ratio of average trail length to number of servers is $13.5/136$ which is close to 10%. This means that a typical access request will be logged at about 10% of proxy servers. Again, whether this satisfies privacy property 2 from Section 2 will depend on specific circumstances. Remember that the fact that a request will typically appear in 10% of the proxy logs, does not mean it can be linked to a given user by inspecting one of those 10% of logs; it does mean that the fact that the query has been issued will be noticeable in 10% of the logs. As noted earlier, this will not be an issue with queries issued often and by many users; it could be an issue when out of the ordinary queries — that could be rather sensitive — are submitted.

Table III lists the threshold values that need to be reached before most requests will be answered from cached copies. It is surprising to note how low the saturation has to be to reach this point. When $\varepsilon = 0.001$ and $\mu = 13.5$, for example, a saturation threshold of only 40% is required. Viewed from the opposite side, once a saturation of 40% has been reached in this case, only one in 1000 requests will have to be sent to an outside server — and every 1000 requests will clearly increase the saturation significantly, causing even fewer requests to be sent to the external server in reality. For all practical purposes, once this level has been reached, under the low internal communication costs assumption of this paper, the system will be as efficient as when a single proxy for all outgoing requests has been used.

As expected, Table III indicates that a shorter median trail requires a higher saturation for a comparable performance; a saturation of 75% is required for $\mu = 6.5$ if one only wants one in 1000 requests to be sent to the external server.

It is simple to tabulate expected saturation levels (σ_n) for various combinations of n, N and α. Assume $\alpha = 0.05$ and

$\mu = 13.5$. Above it was calculated that to have $N \gg \mu$, it is necessary to use $N > 136$. Assume $N = 256$. Then the required saturation is reached after 38 requests. This means the communication cost of using Flocks with these parameters, is that repeatedly used pages will, on average, be transferred at most 38 times from the external server, after which the vast majority of requests will be serviced from a cached copy; some of these 38 requests will, however, be fetched from cached copies. This contrasts with the case of a single proxy where such pages will have to be transferred once. This demonstrates how specific circumstances influence the cost (and therefore the best parameters) for the system: In an organisation where most pages are typically requested more than 38 times, this configuration will incur external communication costs for the first 38 requests. Under conditions where a substantial number of unique pages are accessed, or where pages are typically accessed fewer than 38 times, the increase in cost would be substantially lower; where only unique pages are accessed, costs will be on par with that of a single proxy server. Privacy therefore comes at a cost.

When compared to Crowds, savings will be substantial if pages are indeed often used: In Crowds a single internal proxy is not meaningful and, if a page is requested by a 1000 users, it will have to be retrieved a 1000 times. Flocks, in contrast, will, on average, only have to retrieve the page 38 times. This illustrates that the cost of privacy can be minimised by using an appropriate solution.

A similar calculation shows that, if N is increased to 512, the required saturation is only reached at $n = 77$.

For shorter trail lengths external costs may be lower. For $\nu = 6.5$, an $N = 32$ is large enough (see Table II). However, it has been calculated above that a saturation level of 75% is required to only get one access in 1000 to refer to the external server. A calculation for $\nu = 6.5$ and $N = 32$ shows that this saturation is reached after 22 requests. This lower cost, however, implies that μ/N is larger in than it was for the longer trail lengths. To achieve a ratio of 10% (as was used above), we have to consider $N = 64$. In this case, the required saturation is reached after 58 requests.

3.2 Remaining issues

Flocks is different from most other PETs that have been proposed in a number of ways. The two major differences are the fact that it is intended to be used within an organisation as well as the fact that forensic issues have been taken into account during its proposal.

Most of the directly comparable PETs, such as Crowds, Onion routing, Anonymizer and related technologies have not been developed for use within an organisation. Some, such as Crowds, are simple to apply inside an organisation, but — as has been argued above — come at a relatively high cost when used for specific applications, such as Web browsing. Others, such as Anonymizer, will be of limited use when deployed inside an organisation. Note that Onion routing also uses the notion of (amongst others) an HTTP (Web) proxy (see, for example, one of the earliest papers on Onion routing [Goldschlag et al. 1996]). This proxy, in contrast to Flocks, is one where the HTTP request is injected into the Onion routing network, after which private information is hidden on the network layer. Note the assumption in Onion routing that the network between the user and this proxy is secure, as well as the acknowledgement that this proxy is a vulnerable point in the Onion routing network. (Approaches to minimise this vulnerability have been discussed [Goldschlag et al. 1996]).

A number of PETs have recently been proposed for use inside an organisation. These include Hippocratic Databases [Agrawal et al. 2002] and E-P3P [Karjoth et al. 2003; Ashley et al. 2003]. These PETs address a very different problem from that of Flocks: They protect stored data (typically of customers), while Flocks protects the actions of users (typically of employees). It does, however, seem worthwhile to consider the integration of the two types of technology.

Note that not many PETs have been analysed in this formal manner. The only comparable work that we are aware of, is a discussion on connection setup time, latency and encryption overhead costs in Onion routing [Syverson et al. 1997]. That analysis is based on empirical observations.

Given the fact that Flocks is intended for application within an organisation, ownership and control of data as stored in the logs should be considered. If all log data is centrally controlled, using Flocks will have little value. The notion of *custodianship* has been mentioned above. Different responsible parties should be assigned custody of the various logs — even though the organisation may own all such data. Custody should be managed in a way that allows custodians to be held accountable for the way in which they manage the logs — and, in particular, for the manner in which they control access to their logs.

One technical issue potentially related to custodianship is the question whether it is possible to distinguish between a proxy and an ordinary workstation in a proxy's log. (This would be possible, for example, if the addresses of workstations and proxies were distinguishable in the logs. One possible solution is to configure every workstation as a proxy. Requests from the local workstation are directed at its own proxy. Other incoming requests will look the same, because they will all be coming from a proxy with no indication where that proxy got the request from. In this scenario, it becomes meaningful to consider whether each 'owner' of a workstation should not act as a custodian for the log data on his or her workstation. Using workstations as proxies will also help to address the cost of the equipment: most current workstations have sufficient processing power to be used in this manner.

Surveys of other PETs have been published elsewhere [Lawton 2001; Pfitzmann and Köhntopp 2001; Goldberg et al. 1997; OECD 2002; Seničar et al. 2003; Cranor 1997; Hoffman and Metivier Carreiro 1997]. An architecture that relates the various technologies to one another has been proposed [Olivier 2003].

A number of papers have been published that argue that it makes economic sense to deploy PETs within an organisation [PrivacyRight 2001; Systems 2000; Tivoli Software 2000; IBM 2002]. Most of these papers argue the point from the premise that privacy protection will benefit the organisation's customers and will therefore benefit the organisation over the longer term. It is our contention that a similar argument can be made about protecting the privacy of people inside the organisation (such as employees). Hence, deployment of a system such as Flocks can make economic sense — depending on its potential value compared to its potential cost. This paper used two privacy-related observations and one cost related observation (see Section 2) in an attempt to balance these two issues.

The paper also considered forensics as part of the greater problem. The organisation now is in a position where the private information can be collected — if all the custodians cooperate. Conditions under which such cooperation should be deemed appropriate should be set out in an appropriate policy.

4. CONCLUSION

This paper proposed a random proxy based PET to mitigate the effect of logging on personal privacy. The bulk of the paper investigated the effect of two major parameters (α and N) on the privacy of such a PET. Formulae were derived that can be used to determine the privacy-related properties of the system.

Concrete (numerical) examples of these parameters were discussed next. Although it was stressed that specific choices will depend on specific circumstances, it was shown that the choice of $\alpha = 0.05$ and $N = 256$ is an alternative that would be practical in many cases. This reaches the required saturation level after 38 requests, has an average trail length of 13.5 and every request appears in about 5% of proxies. A cheaper solution could choose $\alpha = 0.1$ and $N = 32$. This reaches the required saturation after 22 requests, has an average trail length of 6.5, but every request will appear in approximately 20% of proxies.

The analysis performed in this paper assumed a negligible cost for accessing internal proxies. It is relatively simple to take such cost into account, by multiplying the average cost of an internal access by the average trail length. This is, however, left for future research.

Future research will also build a simulation of the model to verify the accuracy of the analysis in this paper. Issues that would be tested in such a simulation will include factors that are hard to analyse formally. One example of such a factor is the effect that the number of accesses necessary to reach the required saturation level has on the required saturation level itself. Suppose, for example, that a saturation level is reached where one request in 100 is sent to an external server. On average, request 50 will be sent to this server. However, the 49 requests that precede this request will often increase the saturation level to a level where only one in 1000 will be sent to the external server, which means request 50 will probably not be sent to the external server after all. Hence, it might be possible to use a significantly lower saturation level than the one in 1000 ($\varepsilon = 0.001$) that was used in this paper. This would mainly influence the projected cost as determined in this paper. Eventually it is hoped to construct a prototype to collect further empirical evidence of the effect of the chosen parameters on the operation of the system.

Finally, note that the intended application area for Flocks stems from the analysis performed in this paper and not from any inherent limitation in Flocks. It would be interesting to compare the performance of Flocks in a generic environment with PETs such as Crowds and Onion routing. This is also left for future research.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Research Foundation under Grant number 2054024. Any opinion, findings and conclusions or recommendations expressed in this material are those of the author and therefore the NRF does not accept any liability thereto.

A. SERIES

For the sake of completeness, this appendix derives a few formulae for mathematical series calculations. These series are well-known, but not often encountered in the privacy literature.

A geometric series is a series of the form

$$S = a + ar + ar^2 + ar^3 + \ldots + ar^{k-1}$$

From this it is clear that

$$S - rS = a - ar^k$$

Hence

$$S = a\frac{1 - r^k}{1 - r} \tag{9}$$

A special case occurs when $|r| < 1$ and $k \to \infty$, in which case

$$S = \frac{a}{1-r} \tag{10}$$

Now consider a series of the form

$$T = 0ar^0 + 1ar^1 + 2ar^2 + \ldots + kar^k$$

Here

$$T - Tr = ar + ar^2 + ar^3 + \ldots + ar^k$$

From equation 9 it follows that

$$T - Tr = ar\frac{1-r^k}{1-r}$$

Hence

$$T = a\frac{r - r^{k+1}}{(1-r)^2} \tag{11}$$

More important for the current paper is the case of this series where $|r| < 1$ and $k \to \infty$. Then

$$T - Tr = ar + ar^2 + ar^3 + \ldots$$

From equation 10 it follows that

$$T = a\frac{r}{(1-r)^2} \tag{12}$$

REFERENCES

AGRAWAL, R., KIERNAN, J., SRIKANT, R., AND XU, Y. 2002. Hippocratic databases. In *28th Int'l Conf. on Very Large Databases (VLDB), Hong Kong*.

ASHLEY, P., HADA, S., KARJOTH, G., AND SCHUNTER, M. 2003. E-P3P privacy policies and privacy authorization. In *Proceedings of the ACM workshop on Privacy in the Electronic Society*. ACM Press, 103–109.

CALOYANNIDES, M. A. 2000. Encryption wars: Shifting tactics. *IEEE Spectrum 37,* 5, 46–51.

CRANOR, L. F. 1997. *The Role of Technology in Self-regulatory Privacy Regimes*, Chapter 5. In Daley and Irving Daley and Irving [1997]. http://www.ntia.doc.gov/reports/privacy/privacy_rpt.htm.

CRANOR, L. F. 1999. Internet privacy. *Communications of the ACM 42,* 2, 28–38.

DALEY, W. M. AND IRVING, L., Eds. 1997. *Privacy and Self-Regulation in the Information Age.* US Department of Commerce, Washington, DC, USA. http://www.ntia.doc.gov/reports/privacy/privacy_rpt.htm.

FORTE, D. 2002. The future of computer and network forensics. *Network Security 2002,* 10 (October), 13–15.

GABBER, E., GIBBONS, P. B., KRISTOL, D. M., MATIAS, Y., AND MAYER, A. 1999. Consistent, yet anonymous, web access with LPWA. *Communications of the ACM 42,* 2 (February), 42–47.

GOLDBERG, I., WAGNER, D., AND BREWER, E. A. 1997. Privacy-enhancing technologies for the Internet. In *IEEE COMPCON '97.* IEEE, 103–109.

GOLDSCHLAG, D. M., REED, M. G., AND SYVERSON, P. F. 1996. Hiding routing information. In *Information Hiding,* R. J. Anderson, Ed. Number 1174 in Lecture Notes in Computer Science. Springer-Verlag, 137–150.

GOLDSCHLAG, D. M., REED, M. G., AND SYVERSON, P. F. 1999. Onion routing. *Communications of the ACM 42,* 2 (February), 39–41.

HOFFMAN, L. J. AND METIVIER CARREIRO, K. A. 1997. *Computer Technology to Balance Accountability and Anonymity in Self-regulatory Privacy Regimes,* Chapter 5. In Daley and Irving Daley and Irving [1997]. http://www.ntia.doc.gov/reports/privacy/privacy_rpt.htm.

IBM. 2002. Privacy in a connected world. White paper, IBM. May.

KARJOTH, G., SCHUNTER, M., AND WAIDNER, M. 2003. Platform for Enterprise Privacy Practices: Privacy-enabled management of customer data. In *Privacy Enhancing Technologies: Second International Workshop, PET 2002, San Francisco, CA, USA, April 14-15, 2002, Revised Papers*, R. Dingledine and P. Syverson, Eds. Springer.

LAWTON, G. 2001. Is technology meeting the privacy challenge? *IEEE Computer 34,* 9, 16–18.

OECD. 2002. 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.

OLIVIER, M. S. 2003. A layered architecture for privacy-enhancing technologies. *South African Computer Journal 31,* 53–61.

PFITZMANN, A. AND KÖHNTOPP, M. 2001. Anonymity, unobservability, and pseudeonymity - a proposal for terminology. In *International workshop on Designing privacy enhancing technologies*. Springer-Verlag New York, Inc., 1–9.

PRIVACYRIGHT. 2001. Control of personal information — the economic benefits of adopting an enterprise-wide permissions management platform. White Paper. `http://www.privacyright.com/info/economic.html`.

REITER, M. K. AND RUBIN, A. D. 1999. Anonymous web transactions with Crowds. *Communications of the ACM 42,* 2 (February), 32–48.

RIEKE, A. AND DEMUTH, T. 2001. JANUS: Server anonymity in the world-wide web. In *Conference Proceedings EICAR International Conference,* U. E. Gattiker, Ed. 195–208.

SENIČAR, V., JERMAN-BLAŽIČ, B., AND KLOBUČAR, T. 2003. Privacy-enhancing technologies — approaches and development. *Computer Standards & Interfaces 25,* 147–158.

SYSTEMS, W. 2000. User managed privacy: A new approach for addressing digital privacy and personal information on the Internet. White Paper. http://www.wave.com/technology/PrivacyWhitePaper.pdf.

SYVERSON, P. F., GOLDSCHLAG, D. M., AND REED, M. G. 1997. Anonymous connections and onion routing. In *Proceedings of the 18th Annual Symposium on Security and Privacy*. IEEE, 44–54.

TIVOLI SOFTWARE. 2000. Tivoli Secureway Privacy Manager — controlling access to consumer information. White paper, IBM.

M. S. Olivier, “Flocks: Distributed proxies for browsing privacy,” in *Proceedings of SAICSIT 2004 — fulfilling the promise of ICT*, G. Marsden, P. Kotzé, and A. Adesina-Ojo (eds.), Stellenbosch, South Africa, 79–88, October 2004.

Corrigendum

The calculation of k on page 82 should have looked as follows:

$$\begin{aligned}
k &= \sum_{i=0}^{\infty} i \cdot P(\sigma, n, +i) \\
&= \sum_{i=0}^{\infty} i \cdot (1-\sigma_{n-1})^k (1-\alpha)^{k-1} \left[\alpha + (1-\alpha)(\sigma_{n-1})\right] \\
&= \sum_{i=0}^{\infty} i \cdot [(1-\sigma_{n-1})(1-\alpha)]^k \left[\frac{\alpha + (1-\alpha)(\sigma_{n-1})}{1-\alpha}\right] \\
&= \frac{\alpha + (1-\alpha)\sigma_{n-1}}{1-\alpha} \cdot \frac{(1-\sigma_{n-1})(1-\alpha)}{[1-(1-\sigma_{n-1})(1-\alpha)]^2} \\
&= \frac{[\alpha + (1-\alpha)\sigma_{n-1}](1-\sigma_{n-1})}{[1-(1-\sigma_{n-1})(1-\alpha)]^2}
\end{aligned}$$

M. S. Olivier, “Flocks: Distributed proxies for browsing privacy,” in *Proceedings of SAICSIT 2004 — fulfilling the promise of ICT*, G. Marsden, P. Kotzé, and A. Adesina-Ojo (eds.), Stellenbosch, South Africa, 79–88, October 2004.