Implications of Data Retention Measures for Anonymisation Services

Anonymous Author

Anonymous Institute

Abstract. In an effort to regulate the data retention obligations of ISPs, data retention policies have been legislated. Anonymisation services undermine data retention efforts, as they provide users with a means to masquerade their IP address. In this paper we analyse the obligations specified by the German bill on data retention and present various approaches for implementing data retention measures at single-hop and multi-hop anonymisation services. We determine the amount of storage needed and analyse traceability of users. We show that the obligations in the bill are insufficient to guarantee traceability. Finally, we propose a protocol-dependent data retention tag which ensures traceability while protecting users' privacy.

1 Introduction

For the purpose of law enforcement Internet service providers may be required to uncover the identity of a user given an IP address and a timestamp. In an effort to regulate the data retention obligations of ISPs, data retention policies have been legislated. Anonymisation services undermine data retention efforts, as they provide users with a means to masquerade their IP address. With increasing awareness and usage of the anonymizers, their providers become a target for data retention regulation. During an investigation, law enforcement authorities may approach a provider of an anonymiser with a query q = "Please determine the identity of the user who used your anonymisation service with IP address $a.b.c.d$ on 2008-02-10 at 9:43 am GMT". The anonymiser can only answer this query if the provider has stored detailed records of all served requests – a duty which he will not carry out voluntarily.

The basis for this paper are the ongoing data retention efforts in the European Union. in In 2006 the EU adopted a Directive on data retention (2006/24/EG). Article 1 outlines its purpose as follows

> 1. This Directive aims to harmonise Member States' provisions concerning the obligations of the providers of publicly available electronic communications services or of public communications

> networks with respect to the retention of certain data which are generated or processed by them, in order to ensure that the data are available for the purpose of the investigation, detection and prosecution of serious crime, as defined by each Member State in its national law. 2. This Directive shall apply to traffic and location data on both legal entities and natural persons and to the related data necessary to identify the subscriber or registered user. It shall not apply to the content of electronic communications, including information consulted using an electronic communications network.

The implementation of the Directive regarding internet access can be postponed by member states until March 15th, 2009. The Directive provides only a framework for data retention measures. It is rather generic and does not mention anonymisation services at all. Member states may concretise data retention obligations on a state level, though. The German implementation for Internet access, which is expected to go into force on January 1st, 2009, was designed with anonymisation services in mind. Providers of telecommunication services will have to retain transformation data for a time period of six months. It is well worth studying its implications for anonymisation providers, which is the purpose of this paper. Although our findings are based on a German bill, they are of generic nature and may be applicable to countries with similar regulations throughout the world.

The main purpose of data retention measures in the context of anonymisation services is *traceability*, i. e. to allow law enforcement authorities to uncover the true identity of a suspect, who may have used the anonymisation service to perpetrate illegal actions. Many people consider traceability a good thing and necessary for crime prevention. On the other hand, data retention measures are questionable from a privacy point of view. Consequently, the challenge is to devise data retention mechanisms which are able to allow for traceability while preserving users' privacy. We found that the data retention obligations specified in the German bill are not sufficient to guarantee traceability of users of anonymisation services. Therefore, the purpose of these obligations is questionable.

In the interest of conciseness we limit our analysis to the anonymisation of HTTP traffic. Our results can be applied to all kinds of anonymisation services, e. g. CGI-Proxies based on HTML forms, VPN solutions, Mix cascades and Onion Routing. For clarity we will consider single-hop systems (requests are relayed at a single node) and multi-hop systems (which relay traffic over multiple nodes) separately.

This paper is structed as follows. In section 2 we will summarise the legal obligations specified by the German bill on data retention to ensure a common understanding. The main part of the paper presents various approaches and their implications for the implementation of data retention measures for anonymisation services. First, we will consider simple single-hop anonymizers in section 3 in order to illustrate (1) which level of detail is required by law and (2) which level of detail would be required for guaranteed tracability of users. We will then apply our findings to multi-hop anonymisers in section 4. In section 5 we will show that it is possible to tailor an anonymiser which complies with all legal obligations, but cannot be forced to trace its users based on data retention measures. We summarise our findings in section 6.

2 Legal Basis for Data Retention

From the viewpoint of law enforcement authorities anonymisation services are some kind of public telecommunication service, and, as such, have to meet legal obligations for data retention. Although there are similarities to "classical" ISPs, there are a number of differences:

- anonymisation services hide the original user IP and replace it by their own IP
- there is no physical data link to identify users, but only their originating IP
- there are usually no user databases, but anonymous users only
- free services cannot provide name and address of a user, but only IP addresses as identifiers
- only paid anonymisation services can provide personal data of users

The bill on data retention tries to cover the above differences by treating proxies and anonymisation systems as transformation systems (§ 113 Telekommunikationsgesetz/TKG). While anonymisation systems cannot be expected to store identifying information, which is usually held by the providers only, they carry valuable transformation information. They are therefore obliged to store

- the original ID of the user
- the transformed ID of the user and
- the timestamp of this transformation.

However, most anonymisation services provide a non-bijective transformation for IP adresses. This means, the users of such systems are sharing

a few or even only one common IP address. This is different from regular ISPs, where each user gets an individual IP address. For uncovering certain requests, it would therefore, in most cases, not suffice to store IP address and timestamp of the request. This is an issue that some legislation in various countries, like currently in Germany, might not address correctly.

Information about visited web sites is not allowed to be stored according to German law, neither by anonymisation systems nor by other telecommunication providers. Furthermore, the law does not clearly state whether the data has to be stored for each session (connection to the anonymisation system) or for each request (connection to target server through the anonymisation system). These restrictions and unclarities, as described in the following sections, make it very difficult and almost impossible to implement data retention techniques that would allow to uncover connections unambiguously.

3 Data Retention for Single-Hop Anonymisers

Single-hop anonymisation services replace the IP addresses of their users by different ones, i. e. their own IP addresses. Consequently, providers of such services have to record their usage according to data retention regulations. Concretely, they are obliged to store the input ID and the output ID together with the timestamp of the transformation of the IDs.

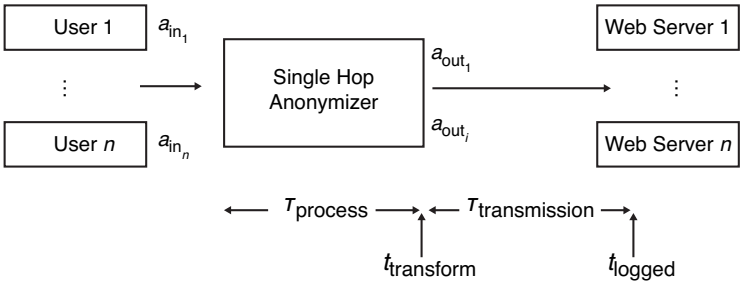

Fig. 1. Model of a single-hop anonymisation service

For the purpose of our analysis we assume that a single-hop system receives HTTP requests on one (or multiple) hosts which act as proxy servers, among them are regular HTTP Proxies, CGI-Proxies (e. g. PHProxy) and VPNs (e. g. anonymizer.com or Cyberghost).

Figure 1 illustrates the single-hop setup. From the viewpoint of the system there are n users with IP addresses a_{in_i}, which send their requests to the proxies. The proxies transmit the requests to the destination servers and forward the corresponding HTTP responses back to the users. From the viewpoint of the destination server, the request originates from an IP address a_{out_i}. Note that $A_{in} > A_{out}$ in most cases, i. e. the number of unique input addresses exceeds the number of IP addresses of the proxy.

Usually, the requests spend some (indeterministic) processing time τ_{process} within the proxy and on the wire towards their destination $\tau_{\text{transmission}}$. Therefore, there is an unknown delay between the transformation at $t_{\text{transform}}$ and the logging of a request at the destination server at t_{logged_i}.

3.1 Recording Input Addresses on Session Level

For session-oriented services (e. g. HTTP proxies which support request pipelining or VPN based anonymisation services), there is a minimalistic and lightweight data retention solution: just record the relevant session-level information. A session may last anything from a few seconds (for HTTP proxies) up to several hours (for VPNs), depending on the individual system setup and user behaviour. If t_{start} and t_{end} denote begin and end timestamps of a user session with the anonymisation service, the provider will store the tuple $(t_{\text{start}}, t_{\text{end}}, a_{\text{in}}, a_{\text{out}})$[1] for each session, meeting the legal data retention obligations (storing input ID and corresponding output IDs together with timestamps). In this scenario, the system does not have to consider the individual HTTP requests, which are relayed during a session. Note that this approach is very similar to the data retention carried out by ISPs.

The storage requirements for this approach are pretty low. Assuming that each of the Unix timestamps and IP(v4) addresses consumes 4 bytes, each log entry requires 16 bytes of storage. Given an average of n_s newly created sessions per second, the service has to provide storage capacity of $16 \cdot 3600 \cdot 24 \cdot n_s = 1\,382\,400 \cdot n_s$ bytes per day. For a service with a daily total of 10.000 sessions this amounts to only 156 kilobytes per day or 27 megabytes in 26 weeks.

From a privacy point of view this solution is the most desirable form of data retention. Only a bare minimum of information is recorded. Personal information – apart from the usage time – is not stored.

While a similar approach is adequate for ISPs, it ultimately fails to meet the purpose of data retention: traceability of the users of the

[1] The applicable time zone information can be stored globally, outside of the session log, as it changes only seldomly.

anonymisation service. Faced with the query $q = (t_{\text{query}}, a_{\text{out,query}}) =$ (2008-10-10 9:43am GMT, a.b.c.d) from section 1, the service provider is not able to uniquely identify one of its users as requested – it can only provide a result of all sessions which match the query. The result set consists of the IP addresses a_{in_i} of all sessions which were in established state at t_{query} and relayed over $a_{\text{out,query}}$ – regardless of users' (in-)activity at a given point in time. Alas, even inactive users contribute to the anonymity group. For increased security, anonymisation services will make sure that outbound traffic uses only a single IP address. Consequently, tracing a request back to its originator is only possible if the number of users at the queried point in time equals one.

Although this approach technically meets the legal obligations, its effectivity is too low to satisfy the needs of law enforcement. Consequently, it is expected that service providers will be forced to store data on a more detailed level.

3.2 Recording Input Addresses on Request Level

There are anonymisation services which operate on a request level by design, e. g. CGI-Proxies, where each address has to be entered in a web form. Data retention on the session level (cf. section 3.1) is not an option for them for obvious reasons, instead they have to log information on a request level. As mentioned earlier, session-oriented services may also be required to store records on a more detailed level.

For this approach an anonymisation service stores the tuple $(t_{\text{transform}}, a_{\text{in}}, a_{\text{out}})$, where $t_{\text{transform}}$ is the point in time when the input address is transformed into the output address. The storage requirements amount to $1\,036\,800 \cdot n_r$ bytes per day, with n_r denoting the number of requests per seconds. In comparison to session-level data retention, the required amount of storage is a lot higher as usually multiple requests are relayed during one session ($n_r \gg n_s$). An anonymisation service with an average of 50 requests per second has to provide storage for 9 gigabytes of retained log files over a period of 26 weeks.

For session-oriented services this approach leads to smaller and thus more precise result sets, because inactive users who are still connected to the service are not included in the result. Nevertheless, this approach is not able to guarantee traceability of all requests. Given the resolution of one minute in a query q (which is standing operating procedure in Germany at the moment), there is a set R_{matching} of $120 \cdot n_r$ matching HTTP requests on average (the requests whose $t_{\text{transform}}$ is included within the two minute interval $[09{:}42{:}30; 09{:}44{:}30[$). The result set $A_{\text{result,in}}$ consists

of all distinct addresses a_{in} which were logged for those requests (where usually $|A_{\text{result,in}}| < |R_{\text{matching}}|$).

The storage requirements of this approach can be further optimised. A typical web browser retrieves a large number of objects via individual HTTP requests. Given the expected time resolution δ of queries it is enough to store one record for a given input address within each time interval δ, i. e. for each given minute only the first request originating from a given IP address is stored.

We have shown that data retention on the request level is feasible. It comes with the cost of storing slightly more information about personal habits of users, though. As long as more than one client is using a single-hop anonymiser concurrently, this approach is not able to trace back a request to its originator.

3.3 Increasing Time Resolution

Apparently, the described ineffectivity of data retention on the request level is mainly due to the poor time resolution used by law enforcement authorities. We will now discuss the expected benefits incurred by increasing the time resolution. Note that the German bill doesn't mention explicit precision requirements for the timestamps at the moment. We have identified two influences on the precision of timestamps, namely clock drift and varying transmission times $\tau_{\text{transmission}}$.

Currently deployed proxy servers are able to log individual requests with sub-second precision. For such high resolutions the drift of the hardware clock becomes an issue. We have observed deviations of up to 1 second within a 24 hour interval. In case authorities investigated the IP address a_{out} of the proxy by analysing log files of the destination server, they have either to make sure that the clock of this server is synchronised, too, or they have to determine the deviation of its clock from the universal time and correct the supplied value of t_{query}. Therefore, even with daily clock synchronisation on proxy servers (which is currently not compulsory to the best of our knowledge), using timestamps with sub-second precision does not make sense.

Furthermore, the transmission time $\tau_{\text{transmission}}$ between the host of the anonymisation service and the destination server may introduce indeterministic timing deviations. For highly loaded anonymisation services or networks transmission times may become non-negligible, i. e. up to several seconds.

Even if it was possible to provide a timestamp with a precision of one second, there would still be n_r requests to consider on average. Although

increasing the resolution of timestamps helps to reduce the size of the result set, it still cannot guarantee traceability.

3.4 Utilising Result Set Intersections

Law enforcement authorities can also make use of advanced techniques which are considered "attacks" from the perspective of privacy researchers. As an example of such an attack, we will describe how they can utilise intersection attacks to reduce the result set.

This approach only works in case an offender visits a given page several times using an anonymisation service. Furthermore, multiple actions of the offender have to be identifiable at the destination server (e. g. because he uses pseudonym or account credentials several times). Authorities could then issue an extended query $q_{\text{intersect}}$ = "Please determine the identity of the user who used your anonymisation service with IP address $a.b.c.d$ on 2008-02-10 at 9:43 am GMT, at 11:05 am GMT and at 3:43 pm GMT".

If the offender used the same IP address for multiple requests, the result sets $A_{\text{result,in}_i}$ for the provided timestamps $t_{\text{query}_\text{i}}$ can be intersected to obtain the final result, which contains only addresses, which were active at all given points in time. Users can prevent this attack by frequently changing their IP address, which prevents linking the individual requests. Anonymisation services which can only be accessed via invariant (pseudonymous) user accounts may be forced to link individual actions using the account IDs instead of IP addresses. Therefore, intersection attacks are especially effective when used against commercial single-hop anonymisation services. Commercial providers can protect their users by accepting anonymous micropayments (i. e. anonymous e-cash).

We have shown that authorities can increase the chance of identifying the originator of a request by linking various information sources. Cautious users can guard themselves against this kind of attacks, though.

3.5 Recording Destination Addresses and URLs

In the previous sections we have excluded destination addresses from data retention. In the following we will discuss the implications of recording them.

Destination IP addresses For increased traceability, anonymisation services may be forced to store the IP addresses of the destination servers. Therefore, they store $(t_{\text{transform}}, a_{\text{in}}, a_{\text{out}}, a_{\text{destination}})$ for each request, which

requires $1\,382\,400 \cdot n_r$ bytes of storage per day (without optimisation). In this case, the earlier mentioned hypothetical anonymisation service with an average of 50 requests per second has to provide storage for 12 gigabytes of retained log files over a period of 26 weeks.

There are organisational and legal issues with this approach, tough: Currently, law enforcement authorities do not provide the destination address in their queries, though. They will have to include this information in their Furthermore, recording destination addresses has been explicitly forbidden in Germany's bill on data retention because of privacy concerns.

This approach is able to considerably reduce the size of the result set. Now, only IP addresses of users requesting an object from $a_{\text{destination}}$ are included. It can be argued, though, that there is still a possibility that several users are accessing different objects on the same destination server, which may frequently happen when various sites are (virtually) hosted on the same physical server.

URLs of web sites An extension of the previouly mentioned concept is the recording of destination URLs in order to further reduce the size of the result set. Note again, that recording URLs is currently not allowed for data retention purposes at the moment in Germany.

URLs require considerable more space than IP addresses. With a conservative assumption of an average length of 50 characters per URL, storage requirements amount to $5\,356\,800 \cdot n_r$ bytes per day. The previously mentioned anonymisation service with $n_r = 50$ requests per second has to provide a storage capacity of 45 gigabytes for the period of 26 weeks (without regarding savings by compression and optimisations) in this case.

The result set will then only contain IP addresses of users which have accessed the same web site within the query time window, which will allow for an exact match in most cases. Traceability is limited for encrpyted web sites (HTTPS), though, where only the host name and port is visible – here the chances of concurrent accesses increase again.

The effectiveness of this approach comes at a high cost. Many URLs contain personal information or sensitive contents (such as search engine queries, session IDs, unencrypted credentials). Storing information of this kind on the hosts of anonymisation services over a period of six months – only for replying to potential law enforcement queries – seems disproportionate and is therefore highly questionable from a privacy point of view.

3.6 Adding Protocol-Dependent IDs

The previous section suggests that high traceability comes only at the cost of impacts on users' privacy. In this section we present an approach to guarantee traceability without the need to store personal information of the users. The only drawback of this solution: it depends on the application protocol. We will demonstrate a possible implementation for HTTP.

The poor effectivity of the already mentioned solutions is due to the missing one-to-one-correspondence between input addresses on the one hand and output adresses, destination addresses or even URLs on the other hand. The proposed solution re-establishes this correspondence by creating a (unique) ID for each request. This ID is included in the HTTP request sent to the destination server, e. g. by means of an additional header `X-Data-Retention-Tag: <ID>`. The anonymisation service stores the tuple $(t_{\text{transform}}, a_{\text{in}}, a_{\text{out}}, \text{ID})$, which establishes an unambiguous link between ID and a_{in}. ID might be generated by determining $ID = \text{h}(a_{\text{in}_\text{i}} + r)$, where h is a hash function like SHA1 and r is a random number to ensure that the destination server cannot link individual requests of the user. As the data retention tag is not logged by the destination server it can only be observed "on the wire" with traffic analysis. While this method is able to guarantee perfect traceability, its voluntary adoption cannot be expected due to its limited use and high complexity.

Instead of proprietary IDs on application level, TCP *source port numbers* might be considered to reduce the size of the result set. They are visible for the destination server and can be used to (at least temporarily) identify requests, because no two requests can originate from the same source port at the same time. Source port numbers are usually incremented and not reused for consecutive connections, which makes it possible to distinguish requests from each other based on them. In contrast to proprietary data retention tags which can be used to uniquely identify a request over long periods of time, source port numbers are ephemeral, though.

Storing additional IDs helps to reduce the size of the result set considerably. With the the proposed proprietary data retention tag perfect traceability can be achieved – at least theoretically.

4 Implementation for Multi-Hop-Anonymisers

Services which relay data for anonymisation over various intermediate nodes are called multi-hop anonymisation services in the scope of this article. This especially includes Tor (onion routing with – in principal –

free choice of intermediate nodes) and JonDonym[2] (static mix cascades). Figure 2 schematically depicts such a multi-hop service with three intermediate nodes.

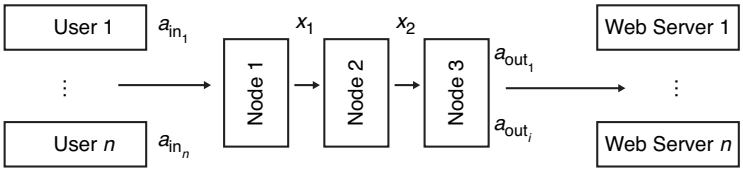

Fig. 2. Model of a Multi-Hop-Anonymisierer with three intermediate nodes

Like single-hop services, the individual routers in a mix network have to store the tuple $(t_{\text{transform}}, a_{\text{in}}, a_{\text{out}})$. Mix cascades always provide the same nodes in the same order. Only the first node has to store a_{in}, the "latter" nodes may to refrain from storing source addresses, as the respective predecessor is statically determined anyway. While in mix networks the retention data of all nodes has to be analysed for tracing, it is sufficient for mix cascades to ask the first node. In mix cacades, tracing therefore seems to be easier as in mix networks.

However, for multi-hop services, the exclusive storage of source IPs and request timestamps is generally not sufficient for tracing (cf. the explanations about single-hop services in section 3). Each proceeding node that forwards the message increases the fuzziness compared to single-hop services: While the fuzziness for $t_{\text{transform}}$ counts δ seconds in the last node, it counts about 2δ seconds in the preceeding node, so that the number of HTTP requests in question, generally spoken, counts $n_{\text{nodes}} \cdot \delta \cdot n_r$ in the first node of a chain.

A better traceability requires the storage of connection IDs that make it possible to uncover the way of a message through the anonymisation service on demand – that is if and only if the operators of all nodes combine their retained data. For this purpose, each two neighbouring nodes use a common ID for each request. The approach is explained using the multi-hop anonymisierers depicted in figure 2. The first node creates an ID x_1 and stores it into the tuple $(t_{\text{transform}_1}, a_{\text{in}}, x_1)$. It transmits x_1 to the second node together with the request. This one creates another ID – named x_2 in the example. The second node then stores the tuple

[2] formerly AN.ON or JAP, respectively

$(t_{\text{transform}_2}, x_1, x_2)$ and transmits x_2 to the third node which, finally, stores the tuple $(t_{\text{transform}_3}, x_2, a_{\text{out}})$.

The intersection explained in section 3.4 is useless concerning multi-hop services, even when newly introduced IDs are used: the IDs are valid for single requests only, and may therefore not be use by mixes to recognize users at later requests. In order to provide an unambiguous link, the last node therefore has to log the destination IP or the reqested URL for all HTTP requests (like proposed in section 3.6). Even an optimisation of memory usage is possible, similar to single-hop services: a burst proxy switched "behind" the cascade or circuit could parse requested web sites and automatically reload embedded objects. No separate data record is needed for these objects any more, as it is clear that they are downloaded from the same user.

The connection IDs allow for an unambiguous tracing of each request up to the first mix, which has stored the corresponding source address. This procedure may be equally well implemented into mix nets and mix casades. In general, these IDs do not have to be newly introduced into systems, as they already implicitly exist in order to allow for the communication of users with the individual servers of the net or the cascade, respective.

5 Compliant Anonymiser with no Traceability

If multi-hop anonymisation services are legally obliged to make the way through their system traceable by using IDs, they, at the first glance, provide only a small security gain compared to single-hop aonymisers against "law enforcement attackers". The fuzziness in the result set of a request, q, is comparable for both architechtures. But the true strength of these systems lies in their distribution, that means that single hops in a chain may be outside the influence of a state enforcing data retention. They then do not have to create or provide any recorded data. Moreover, the user ID in commercial single-hop services allows for very effective intersection attacks on the stored data, which are, in this form and comparable effectiveness, not possible in multi-hop systems.

It is worth to mention that the german law excludes "'Content and data about requested internet sites" from retention (§ 113a Abs. 8 TKG). This specifically includes URLs of requested web sites. In section 3 we have shown that even with a Single-Hop-Anonymiser, an unambiguous tracing is not possible, if only source and destination IP addresses, but not URLs or IP addresses of destination servers are stored. We now may

"abuse" this fact to create a very special multi-hop service based on URLs that falls under the mentioned law, but anyhow does not need to store connection IDs.

This hypothetical service consists of a loose network of web-form-based proxy servers that solely act on HTTP requests, (similar to CGI proxies). Each of these servers may work all alone, but users may also call another form-based proxy through them, and thereby build chains of form-based proxies like this:

URL = http : //www.node1.com/www.node2.com/www.node3.com/http : //www.url.com

where `www.url.com` is the web page requested by the user. This method may lead to quite a high protection against data retention, as, with the lack of ID chains, the proxy servers cannot uncover the way through the network. They cannot be forced to exchange connection IDs, as there are no implicit IDs between the servers. Unfortunately, the servers itself can log the whole user traffic if they like to, and the first server can see request, response and the user IP. However, the difficulty for law enforcement agencies is to uncover the first server in the chain.

6 Summary and Outlook

We have shown that a minimalistic approach to implementing data retention in anonymisation services is not sufficient to guarantee traceability in all cases. Service providers can guarantee traceability by introducing additional IDs, but implementations are mostly proprietary and therefore difficult to deploy. Another option is to store destination IP addresses or even URLs, but this has dire consequences for users' privacy.

Furthermore, we have demonstrated that the obligations specified in the German bill are so imprecise that it is still possible to design and deploy anonymisation services which comply with the law but protect their users' identity.

In the future, anonymisation services might be obliged by law to ensure traceability for their users – as ISPs in the EU are required already today. This would force anonymisation services to ensure a one-to-one-correspondence between the IP addresses of their users and their own IP addresses. Obviously, this is only possible if such services have a large enough pool of IP addresses. While this requirement seems impossible to meet with IPv4, it might be feasible with the IPv6 address space.

References

1. http://www.jondos.de
2. http://tor.eff.org
3. Stefan Köpsell, Rolf Wendolsky, Hannes Federrath: Revocable Anonymity. in: Günter Müller (Ed.): Proc. Emerging Trends in Information and Communication Security: International Conference, ETRICS 2006, Freiburg, Germany, June 6-9, 2006, LNCS 3995, Springer-Verlag, Heidelberg 2006, 206-220.
4. Ian Goldberg, David Wagner: TAZ Servers and the Rewebber Network: Enabling Anonymous Publishing on the World Wide Web. First Monday 3, 4 (August 1998).