# Private Yet Abuse Resistant Open Publishing [*]

(Position Paper)

George Danezis
K.U. Leuven, ESAT/COSIC,
Kasteelpark Arenberg 10,
B-3001 Leuven-Heverlee, Belgium.
George.Danezis@esat.kuleuven.be

Ben Laurie
The Bunker Secure Hosting,
Shepherds Building, Rockley Road
London, W14 0DA, United Kingdom.
ben@links.org

## Abstract

We present the problem of abusive, off-topic or repetitive postings on open publishing websites, and the difficulties associated with filtering them out. We propose a scheme that extracts enough information to allow for filtering, based on users being embedded in a social network. Our system maintains the privacy of the poster, and does not require full identification to work well. We present a concrete realization using constructions based on discrete logarithms, and a sketch of how our scheme could be implemented in a centralized fashion.

## Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection; D.4.6 [**Operating Systems (C)**]: Security and protection; H.3.5 [**Information Storage and Retrieval**]: Online Information Services

## General Terms

Security

## Keywords

Open publishing, abuse resistance, spam, social network, filtering

## 1 Introduction

A definite trend for news services using the World Wide Web (WWW) has been a shift from web site editors providing information on their sites to allowing users to post comments, ratings or full stories. This ability is often abused through off-topic posts, postings that contravene editorial policies, or plain spam (unsolicited, commercially motivated communications). Methods for dealing with such behavior can be based on the content posted. These require constant monitoring of posts, which is quite labour intensive. An alternative approach is to filter posts based on the poster's identity. It is believed that this provides better results since only a minority of posters are persistent abusers. This approach has been difficult to implement due to the lack of a reliable identification infrastructure on the Internet, and the difficulty of creating ad-hoc ones. Furthermore freedom from abuse, to maintain quality of content, must be considered in tandem with the privacy of the posters, who might be discouraged from posting if they are to be fully identifiable.

[*]This work is partly supported by a grant from the FWO and the AEOLUS EU project.

In order to provide a privacy-preserving yet robust solution to the abuse problem we make some fresh, yet realistic assumptions. We shall assume that users, wishing to contribute to an online resource, are somehow introduced into the system by an existing user, forming an 'introduction graph' (first proposed by Lesniewski-Laas *et al.* [15]). The security of our scheme is based on the fact that it is costly to fool many real-world honest users and therefore there is a bound on the number of independent honest users an abusive node can be connected to. Similar structures have also been proposed before the peer-to-peer paradigm, to secure certificate distribution in Wright *et al.* [42] and stream authentication in Song *et al.* [35].

Our key intuition is that we can use the graph path to the abusive node to extract persistent labels on which one can filter out content coming from this user. There is no need to actually get the real world identity or network address of the abuser, or even the full or accurate path in the introduction graph – indeed we make sure that there always exists some degree of plausible deniability [33] concerning the actual contributor. This ultimately guarantees the privacy of posters.

## 2 The abuse filtering problem

Well before the advent of the Web, collaborative news, reporting and discussion was taking place on-line through mailing lists and Usenet newsgroups. We shall first present how these systems coped with abuse.

### 2.1 In the beginning was email…

Mailing lists are 'broadcast' email addresses which expand a received message to all the list's subscribers. Abuse control is performed by list moderators that can filter messages based on their content or the identity of their sender. A usual configuration is to only allow list members to directly post to the list, whilst moderating posts from non-subscribed users (though some lists moderate all postings while others allow anyone to post). Subscribers' email addresses are often checked when they subscribe, using a simple challenge-response mechanism (phone numbers or physical addresses may also be checked in the same way). If a subscribed user becomes abusive (a judgment that is up to the moderator) he can be unsubscribed without his consent. The key assumption, that the originator of the messages can be reliably recognized as a member or non-member of the list, is fragile. There is no robust identification procedure for email senders; email spoofing is both possible and common. Many list managers also employ off-the-shelf spam filtering software to rid the list of unsolicited messages. This provides limited protection against adversaries, that target the list for disruption, by resubscribing under different identities and spoofing messages. This phenomenon is a variant of 'trolling', i.e. making deliberately provocative statements in order to start a 'flame war', and so far mostly social mechanisms have been employed to minimize its impact on an on-line community [39].

Usenet newsgroups [26] provide on-line forums, grouped under a certain number of categories. Originally there were only eight root categories, and creating a group under any of these was subject to a cumbersome process (voting, etc). The *alt.** category allows for a much easier newsgroup formation, and has hosted many groups discussing controversial subjects (sex, drugs and rock-and-roll being only some of them). Usenet groups have been the subject of a lot of spamming and specific technical measures have been fielded to raise the quality of their contents. Only a minority of them are moderated, in which case abuse prevention is done as in mailing lists. The main mechanism for deleting posts is issuing a 'cancel message' deleting a post. Originally everyone was allowed to issue cancel messages, but this mechanism was itself abused, leading to censorship, and nowadays only server administrators and trusted third parties (that perform spam filtering and cross-posting filtering) are allowed to cancel messages. This is again ineffective against persistent disruptors that can craft their messages to get through the controls. Note that through services like Dejanews (which is now Google groups [3]), newsgroups are available for reading and posting through a Web interface.

A handful of insights are already emerging from the description of the pre-web publishing systems. First of all **both access to publishing *and* abuse prevention have to be considered as parts of an effective censorship resistant system**[1]. Limiting the ability to disseminate information can clearly be used for censorship, yet flooding users with irrelevant or inaccurate information (purposefully or not) both lowers the value of the overall news or discussion feed and increases the cost of getting the valuable information. We see that the abuse of the publishing medium can be an effective tool to prevent genuine stories and views from being given the attention they deserve.

This issue is the subject of great controversy in on-line communities that attempt to be inclusive: attempts by any group of people to rid the forum from abusive material is branded (often by the trolls themselves) as 'censorship'. Yet this debate provides us with a further interesting insight: **what constitutes abuse is subjective, yet there are clusters of users whose views coincide on this matter**. Our solution therefore can be applied to satisfy at least one such cluster, by allowing for the filtering of material that does not interest them (since the material is not what they expect from the on-line forum) [2].

[1]As first noted by Richard Clayton.

[2]Note the value-free language, that allows for minorities of posters, or those who are commonly considered abusers, to just create a separate feeds with posts that interest them

Finally we observe that there are satisfactory solutions for dealing with non-targeted abuse, such as spam. Therefore **we shall concentrate on disruption by adversaries that target specific on-line communities** and are determined to lower the quality of the overall news feed or discussion. This phenomenon is often recognized as 'trolling' [39], but can also be an effective tool for information warfare (as it is obvious by browsing forums dealing with controversial issues).

## 2.2 ...then came the Web.

The World Wide Web [21] as introduced in the early nineties very much embodied a publisher/reader distinction. Servers hosted the content of the publishers and clients were accessing it to read it. Interactive services were implemented early, often through the CGI [37] interface, to provide services to allow queries on databases, but only infrequently to change the actual content of the sites.

In the late nineties a new paradigm emerged, which has been named 'open publishing'. Web site would allow users to contribute comments and stories. The most widely known examples, and trend setters, are Slashdot [34] (a news forum on technology and policy issues), Indymedia [1] (providing alternative news feeds from 140 groups around the world), and Wikipedia [2] (a collaborative encyclopedia project). In its purest form 'open publishing' can be defined according to Matthew Arnison [5] as:

> "Open publishing means that the process of creating news is transparent to the readers. They can contribute a story and see it instantly appear in the pool of stories publicly available. Those stories are filtered as little as possible to help the readers find the stories they want. Readers can see editorial decisions being made by others. They can see how to get involved and help make editorial decisions. [...]"

Different websites implement this vision in different ways. Slashdot allows comments and rating of articles, but does not make its internal editorial process transparent. Indymedia attempts to implement the full policy, but fails in terms ease of access to editorial decisions, and filtering is rather heavy to maintain quality. Wikipedia also attempts to adhere to open publishing by making both content and editorial decisions completely transparent. They all suffer from trolling and abusive postings.

Aside from news and knowledge sites, contribution based publishing has become the core of a couple of other web paradigms: these are web logs (also called blogs) and wikis. Weblogs are personal or communal diaries, often allowing for comments from complete strangers. Wikis are free-form pages that allow anyone to edit them, using a simplified markup language.

A common difficulty that all open publishing systems encounter, when it comes to filtering abuse, is the lack of user identification. The Internet only provides weak clues that could be used to associate different posts with each other, and possibly with an abusive poster. Often this is seen as a good thing, since anonymity might be required when discussing or reporting on controversial topics.

Slashdot requires users to login and authenticate a pre-registered account to contribute comments and ratings. This is rather weak, since it is possible to register many accounts which can be used as different identities. Wikipedia allows non authenticated users to contribute material, but has a policy of black listing particular IP network addresses which are the source of abuse[3]. Indymedia refuses to ask for identification in order to contribute articles, standing firmly by the principles of anonymous political speech.

As John Douceur described in his work on the sybil attack [20] in peer-to-peer systems it is difficult to avoid an adversary that masquerades under multiple identities, and thus appears to be many different people. This difficulty is at the core of performing identity based filtering in open publishing systems. Different approaches have been used to combat the sybil attack, two of which are subnet black listing and CAPTCHAS. In the case of subnet blacklisting it is assumed that the adversary can modulate their IP address but only within a particular subsection of the IP space. This belief cannot apply to a determined adversary that can (for a small fee) buy time on one of the many available bot-nets [22], spanning most of the IP space. CAPTCHAS [40, 6] are deformed strings of characters which are difficult to parse automatically. They are presented to the user to make sure upon registration that a real human is indeed performing the operation, and not an automaton. A typical attack against them is to relay the challenges to other users, or to simply pay others to solve them (i.e. relay them somewhere where labour is cheap). These approaches are not suitable to defend against abuse from a determined adversary. Furthermore, solutions based on Public Key Intrastructures [18] or Single Sign On (such as Passport and Liberty [8]) do not seem to be widely deployed (although OpenID [4] is making some progress). They are also overly privacy invasive for the purpose of filtering abusive posts on open publishing systems, and directly conflict with the anonymous speech ethos that many such sites advocate.

As we will see below, when we describe our solution, failure to solve the abuse problem is related to the assumption that the set of users has no structure at all. We shall assume the pre-existence of a social network in which users are embedded, and explore how we can reduce posting abuse based on this.

## 3 Our Solution

At the heart of our solution to the abuse problem is leveraging the existing real-world or virtual social ties amongst users, and building a loose labeling system based on them. Assuming that there is a cost associated with creating local social ties we will see that an adversary will also find it costly to associate a completely different label to different posts. We can use this insight to help filter abuse originating from a small number of users.

The first stage of our protocol is the **introduction**. Assume that Alice wants to post to a certain service to which her acquittance Bob is already introduced. She simply asks Bob to introduce her to this service. The person who created, or controls, the service is by default the first one to be introduced, and we shall call them Root. Conceptually they are at the root of the introduction graph representing users introducing each other to this service.

Being introduced into the system automatically gives Alice the right to introduce others. We will see that this is crucial to allow plausible deniability of the identity of posters. We assume (and shall provide a set of incentives to reinforce this) that it requires effort for an adversary to be introduced to the system by multiple introducers. At the same time an adversary, once connected to the introduction graph, can introduce a large number of dishonest nodes. Furthermore they can introduce themselves in such a way as to not arouse suspicion.

[3]The fact that the Tor anononymizing network [19] addresses were included in this list was the initial staring point of this work.

Note in our simplest scheme there is little reason to centralize any information about the introduction graph. As we will see there is a need to keep local information, relating to the person you are introduced to, and to those you have introduced. In fact, Bob should be able to prove to third parties that he has connected Alice. We stress that there is no need for Alice's real-world identities to be known (Bob can actually ignore her real-world identity). Further information disclosed as part of our protocols is explicitly described later, when we present possible realizations of our scheme.

The most common operation that Alice will perform is to **post messages** to the web site. In order to do this she has to prove a few things: She needs to prove that there is a path connecting her to Root in the introduction graph, and disclose the identity of the first node on the path from Root to her. Then she should provide the message to be posted and enough additional information to perform the 'taking responsibility' steps, described below, if necessary. Alice's post is then published by Root.

A user, let's say Charlie, has the ability to **object to a message** posted. In line with the requirement for openness, we require Charlie to provide in clear and in public his objection and the full path in the introduction graph that connects him to the Root. It is rather important to fully identify those who initiate or perform editorial procedures so that these are not themselves abused.

The asymmetry between posting and objecting is intended: when objecting a user must reveal up front their full path to Root, for anyone to judge the objection (posters only have to prove they are connected through some path). One could design a system in which only limited information is provided when objecting, unless another user objects to the objection. This could lead to infinite recursion. Instead of objecting to the objection, one could also take responsibility for the controversial article. It would still be difficult to automate this: it is easier to filter out frivolous and persistent objectors when presented with their full path to Root. It is in line with the spirit of open publishing to provide a fully transparent editorial process, therefore we opt for simply requiring full identification when objecting.

We will call the main abuse control mechanism we propose **taking responsibility**. The aim is not to retrieve the identity of the poster of abuse, but to associate a label with the message that is related to this identity in such a way that it is difficult for an abuser to frequently change it completely.

Upon an objection to a post being registered there is a public call for any user to take responsibility for it. The full post and the full identity of the user that has objected to it (his path to Root) is provided to allow users to make the decision of taking responsibility or not. Any user that considers that the message is not abuse can step forward and take responsibility for it, not just the original author. We explicitly disallow 'blind' tracing or taking responsibility: this means that the objector must have seen the message to object to it, and someone who wishes to take responsibility for it must have seen the original message and objection.

In practice taking responsibility means that the user, let's call her Denise, agrees to associate her full identity (path to Root) with this post. As a result the controversial post is labeled with the identity of Charlie, who objected to it, and Denise, who has taken responsibility for it.

In the case that no user takes responsibility, the first user on the path from Root to Alice is asked for the identity of the user that connects them to Alice. If that user fails to comply then they are automatically assumed to be taking responsibility for the controversial post, and their full identity is associated with it. Thus users that connect Alice to Root start revealing the path to Alice, unless they accept responsibility for the post.

Note that users on the path to Alice are free to lie and claim that a different link leads to the author of the message (the ability to lie is limited to actual or fictitious links to them, so that absolutely random users cannot be framed). These mechanisms, along with the fact that 'Alice' could be a pseudonym, guarantee that our protocol can never give any strong evidence about the true originator of the post. It also does not guarantee strong anonymity: some information is leaked about the author of the message unless another user takes responsibility for it. In particular part of the path to the sender will be revealed, but there will always be uncertainty about whether some of the nodes taking part in the 'tracing' have lied to hide the true sender of the message[4].

This mechanism always provides a path to Root associated with the controversial post. This path, along with the path of Charlie to the Root, becomes the label associated with the post that allows for filtering.

The ultimate purpose of our system is to allow users to **filter** messages. The exact criteria of the filtering have to be left to website administrators or, even more in the spirit of open publishing, end users. Our key contribution is that, when it comes to controversial articles, they are provided with a label containing the full path of the objecting user and the user that has ultimately taken responsibility. The prefixes of these paths that are closer to Root should provide a stable enough string (i.e. the adversary will find it difficult to manipulate and modulate it) to be a robust filtering criterion against persistent abusers.

Our system also supports many variants of collaborative filtering: users can publish their blacklists (containing prefixes that usually generate abuse), for others to use. Social choice mechanisms, such as elections, can also be applied, but care has to be taken for sybils to not be able to 'stuff the ballot box'. One could use the path of the different users to the Root as a mechanism to make sure that they are independent and not related. Users with different path prefixes to Root are less likely to be controlled by a single adversary. Guaranteeing that such elections are not manipulable, given the sybil assumption, is an interesting but separate problem.

Users should, finally, have the ability to **disconnect** other users they have introduced to the service. It is clear from the mechanism described above that abuse originating from introduced users, such as Alice, also associates the name of the introducing user, say Bob, with the label of the potentially abusive message. Unless some other user accepts responsibility, Bob is also able to see the messages originating from Alice, and might also decide that their relationship should not be maintained. In this case Bob can sever the link with Alice, at which point she will need to find another node to be re-introduced into the system. Alice can also disconnect from Bob at any time.

[4]Allowing introducers to 'frame' introduced nodes, and undetectably lie about abusive messages having originated from them, provides incentives to connect through trusted and non-abusive nodes.

## 4 Interactive realization

We show how to implement our proposed abuse resistant publishing mechanism without the need for a central trusted authority (we sketch in Appendix A how a centralized implementation could work). Our construction will rely on ElGamal encryption [23], the ability for anyone to re-encrypt ElGamal ciphertexts without knowing any private keys [25], and simple zero-knowledge proof of knowledge of discrete logarithms [9]. These proofs can also be used to construct signatures, or a standard signature scheme based on discrete logarithms can be used [36]. On the downside, we require the users linking Alice to Root in the introduction graph to be on-line and participate in the protocol for each post, even those that are not objected to. Our construction should withstand passive adversaries (also known as honest but curious).

We shall assume that all communications take place over anonymous channels, and that the identities of the participants in all protocols are pseudonymous (i.e. not linked to a real-world identity or any identity in another system). Furthermore, communication between honest users takes place over authenticated (using their pseudonymous identities) and confidential channels. These encryption or authentication mechanisms layered above do not provide non-repudiation, meaning that it should not be possible for Bob to prove that Alice sent a certain message, unless we explicitly use signatures to provide this property. Most hybrid stream encryption mechanisms, such as TLS [17], have this property. Off-the-record communication channels [7, 32] guarantee plausible deniability and forward secrecy and would, therefore, be a perfect choice.

ElGamal encryption security relies on the difficulty of computing discrete logarithms modulo a large prime. Bob, the receiver of messages, chooses a secret key $\text{Priv}_B = x$, and computes a public key $\text{Pub}_B = (g, g^x)$, that he gives to Alice. Alice encrypts a plaintext message $M$ by choosing a random nonce $k$ and computing the ciphertext $(g^k, g^{xk} \cdot M)$. Note that any party can re-encrypt the ciphertext, given only the public key under which it is encrypted, by multiplying in some factors: given a fresh nonce $k'$, the new ciphertext will be $(g^{k'} \cdot g^k, g^{xk'} \cdot g^{xk} \cdot M)$. Universal re-encryption [25] can also be used to do away with the need to know the public key of the receiver to re-encrypt but it does not provide any efficiency or security improvements for our scheme.

We implement the different phases of our scheme in the following manner:

**Introduction.** Bob is introducing Alice into the system. Alice and Bob perform a key exchange that leads to a shared key $k_{ab}$ that they use to protect the confidentiality and integrity of all subsequent communications between them. They also generate a 'link certificate' that can be used by either parties to prove that there exists a link between them.

$$A \leftrightarrow B : \text{Signature}_{AB}(\text{Link}, A, B, H(k'_{ab})) \tag{1}$$

We assume there is a good binding between the names of Alice and Bob and their respective public keys, otherwise the latter should also be included in the signature. The hash $H(k'_{ab}) \equiv H(H(\text{Revoke}, k_{ab}))$ of a derivative of the shared secret $k_{ab}$ can be used to revoke links. If Alice or Bob make $H(\text{Revoke}, k_{ab})$ public the link certificate is considered to be no longer valid. Bob also provides to Alice all the 'link certificates' that link him to Root.

**Posting.** Alice wishes to make a post on Root's service. She first generates a fresh public key $(g, g^y)$. Then she gives Bob (over an authenticated and confidential channel) an ElGamal encryption of the fresh key encrypted under Root's public key $(g, g^x)$:

$$A \to B : (g^k, g^{kx} g^y) \equiv c_{AB} \tag{2}$$

Bob will pass it on to Fiona[5], who is connecting him to Root after having re-encrypted the ciphertext with a fresh nonce $k'$:

$$B \to F : (g^{k+k'}, g^{(k+k')x} \cdot g^y) \equiv c_{BC} \tag{3}$$

Bob also stores in a table the following information:

$$\left[A, F, H(c_{AB}), H(c_{BC}), (g^{k'}, g^{xk'})\right] \tag{4}$$

Eventually Root will receive a ciphertext $(g^{\Sigma k_i}, g^{x \Sigma k_i} g^y)$, and use its secret $x$ to decrypt it and recover the temporary key $(g, g^y)$. The fact that the message arrived is proof that there is a chain in the introduction graph between the creator of this key and Root. Root stores in a table the ciphertext, the key and the final node that delivered the message.

Alice then sends her message anonymously (as always) to Root:

$$A \to \text{Root} : g^y, M, \text{Signature}_y(M) \tag{5}$$

Root posts the message on the service, and stores the signature.

**Object!** Any user, lets call them Charlie, upon seeing the message $M$ published can object to it. They need to provide Root with a signed objection containing their full path in the introduction graph.

$$C \to \text{Root} : \text{Signature}_C(\text{Object!}, M, \text{'Intro. links to Root'}) \tag{6}$$

Root makes the objection public, and asks if any user would take responsibility. Users can do this by sending their full path in the same way as Charlie did. In this case their full address and the address of the objecting user are associated with the message by Root, to allow users to perform filtering.

**Taking Responsibility.** If no user has taken responsibility for the post when the objection was broadcast, the process of assigning responsibility starts. Root asks the user Fiona, that has provided her with the ciphertext $(g^{\Sigma k_i}, g^{x \Sigma k_i} g^y)$, for the next user down the chain. To do this, Root has to prove that the public key associated with the signature on the offending message came indeed from her ciphertext.

Given the offending key $(\alpha, \beta) = (g, g^y)$ and the ciphertext $(\gamma, \delta) = (g^{\Sigma k_i}, g^{x \Sigma k_i} g^y)$, and root's key $(\alpha, \varepsilon) = (g, g^x)$. Root has to show in zero-knowledge that she knows $x$ such that:

$$\alpha^x = \varepsilon \wedge \frac{\delta}{\beta} = \gamma^x \tag{7}$$

This can be easily achieved using standard schemes [9] by proving that Root knows $x$ such that $g^{\Sigma k_i}, g^{x \Sigma k_i}$ and $(g, g^x)$. At this stage Fiona is convinced that the offending post $M$ was indeed signed with the message/key she transported.

Fiona has a few choices at this stage: she can fully cooperate with Root, and provide her re-encryption factors $(g^{k_F}, g^{xk_F})$ and previous user Bob. She also has to prove that she is connected to Bob, by providing their 'link certificate'. Root would have to prove to Bob that the complaint concerns a user linked by him, and the process shall continue.

Alternatively she can construct two different re-encryption factors $(g^{k'_F}, g^{xk'_F})$ and pretend that the message came from another one of her links, or even a fictitious link. Neither Root, nor anyone else should be able to find cryptographic evidence to contradict Fiona.[6]

Queried nodes also have the option of stating that they have never sent such message, or that they cannot remember it. That might well be true if a user closer to the Root has lied about the origin of a message. Queried users can also take responsibility for the post (although they already had a chance when the objection was first raised).

In any case Root attaches the 'link certificates' of all the users that have been queried to the post, so that they can serve as labels for filtering. The finally published article looks like:

$$\left[M, \text{Labels} : \text{Object!}_{C \ldots \text{Root}}, \text{Resp.}_{\ldots, B, F, \text{Root}}\right] \tag{8}$$

**Filtering** Users, or Root itself, can construct rules based on the labels of messages. Nodes that are closer to Root are more likely not to be controlled by abusers, while nodes down long chains are more likely to by Sybils. In any case, sub-spaces of the introduction tree that generate a lot of abuse can be identified, through their common branch, and filtered. Since taking responsibility involves users on the introduction chain seeing the message and objection, they may also decide to unlink the part of the introduction tree that is generating too much abuse.

Let us provide a bit of intuition about the security, but also limitations, of this scheme. The Introduction generates a certificate chain, that can be used by Alice or Bob to prove that there is a link between them. This certificate is necessary to prove that a post could have originated from Alice, but also necessary for Bob to be able to blame Alice for a post she did not send.

The Posting phase is designed to maximize plausible deniability. The session public key from the poster bubbles up towards root and gets re-encrypted. It is for the security of this step that we must assume that the intermediaries are curious but honest: if any of them show the encrypted key to Alice she can decrypt it and correlate it with the final ciphertext that she receives. Any of the intermediate nodes is also in a position to silently drop the message, denying in fact service to the poster. Are these security limitations important given our security model? The poster has willingly connected to a

[5]Note that if Bob is modifying messages he can confirm that the ciphertext is the encryption of $g^y$ by dividing the plaintext out and multiplying in his own. Confirmation is possible since Root simply publishes all received plaintexts (and therefore acts as a decryption oracle). In order to protect against such (non-passive) adversaries we would need to randomise the plaintext further before encrypting, so that Bob cannot 'guess' the plaintext. Further modifications to the proofs during the 'taking responsibility' phase would also be needed.

[6]It is trivial to extend this scheme to prevent Fiona blaming one of her honest links. We could require each message in the posting phase to be signed, and in the taking responsibility phase a valid signature attached with the ciphertext presented to Root. Fiona would then not be able to compute a valid signature on behalf of an honest link, but could still pretend the message came from a ficticious link. We prefer to allow her to frame honest links to give incentives to connect only to trusted users.

user, and been introduced by a particular chain of users to the system. The fact that those users can both decrease (not compromise completely) the anonymity of the post, as well as deny service may in fact be acting as a worthy incentive to chose the introducing node with care!

The Taking Responsibility step is the most challenging security wise. Root must prove that the key used to encrypt the post originated from a particular chain to start 'tracing back' the path to the author. Other nodes when solicited for the re-encryption coefficients can, on the other hand, lie and claim that any connected node is the originator of the message. This relies, technically, on the fact that they can provide 'fake' re-encryption factors leading to a fake ciphertext: the channel then between them and the nodes they introduce must provide plausible deniability so that there is nothing to contradict their claim.

## 5 Discussion

First we need to discuss why this scheme provides resistance to abuse. Our assumption has been that it is difficult for the adversary to 'fool' many honest users in the introduction graph to connect him. Therefore it will be expensive to acquire a lot of differently prefixed paths to Root that could be used to unlinkably abuse the publishing medium. If the abuser connects a large number of sybils to a particular subtree of the introduction graph, then the prefix path that connects it to Root can be used to consistently filter out the posts. This means that rules can be made to, for example, hide all posts (or discard objections) coming from that subtree of the introduction graph. Given our assumption, it takes an adversary an amount of effort linearly proportional to the volume of abuse sent (with a large constant overhead involving socially engineering new people), so the Sybil attack is defeated: No matter how many sybils are connected cheaply to the same subtrees, the amount of unfiltered abuse should not increase much.

But what are the incentives for users to exercise care when introducing others into the system? Our assumption that users are embedded into a social network should be supported by the right set of incentives. As discussed in the overview of our scheme if an introduced user misbehaves, the label used to filter the message contains the identity under which the introducer posts. Therefore, in order to avoid their messages being filtered out, introducers have incentives to connect the smallest possible number of abusers. Similarly if they perceive that a subtree they are connecting to the system is generating a lot of abuse and objections, they have incentives to disconnect it.

Users trying to connect to the system to post also have incentives to get introduced by non-abusive nodes. As before this decreases the likelihood their posts would be filtered when objected to. Furthermore, we allow introducing nodes to lie as to the exact sender of the message: therefore a malicious introducer can blame particular abusive messages on their downstream links. This provides even more incentive for the user to have some degree of trust in the introducer.

A difficult question relates to accessibility rather than security: can we assume that all users wishing to post material can find an introducer? First sociologists provide us with an answer, since many studies show that social networks not only have a low diameter, but are also efficiently navigable [38, 27]. Recent systems such as the blog community livejournal [29], the social networking site Orkut [30] and Google's email service gmail [24] were originally accessible by invitation only. All three managed to gain considerable size without allowing for the public to register, but instead requiring an introduction from an existing member. Although this is a positive indication that invitation-only systems will scale and be inclusive, a contributing factor to their success is their generality. It might prove difficult to build an 'introduction only' community based on a very narrow interest group. Therefore it might prove valuable for sites to federate and use the abuse control infrastructure in common. We will discuss this briefly in section 7.

## 6 Preliminary Evaluation

The key objective of our work is to establish persistent labels that can be used to filter out content that is not of interest to particular users who consider it 'abuse'. We advocate using the full introduction paths to the Root of a group both in case a user wishes to object to a post and in case they wish to stand by its content, by taking responsibility for it. In this section we have performed some simple experiments to establish if such labels can effectively be used to infer users' preferences and perform filtering.

In any experiment involving introduction graphs one has to make some assumptions about its social structure.

We use the model introduced in [12] and consider nodes with two views or preferences for content namely 'Blue' and 'Red'. In the context of this work content posted by a node of a particular color would be supported by nodes of the same color (that are happy to take responsibility for it), and objected to by nodes of the other color. Note that the experiments maintain the neutrality of our model, but not branding one preference 'good' and the other 'bad', but simply considering them as different.

We consider a set of nodes, half of them with blue and the other half with red preferences. One of the nodes is selected at random to be Root, and introduce other nodes to the system. All nodes are allowed to introduce up to five (5) other nodes. The key security assumption that our work relies on is that it is relatively rare for a node of a certain color to introduce a node of the other color, and it takes such a child a lot of effort to accomplish this. For this reason we assume that a node only introduces a node with different color with probability $1/10$, and otherwise introduces a node of the same preference. Nodes introduce each other, given these constrains, until all nodes are part of the system.

Once all nodes are introduced, posts start being generated by random nodes. Each post is objected to by a random node of the other color, and supported by a random node of the same color as the poster. We consider a user that looks at such a stream of labeled posts and tries to decide what to filter in the future. To do this it attempts to assign colors to the different principals in the system: every time it sees a post it likes being supported by a node it assigns the node, and all the nodes on its path to Root, the same color as itself, otherwise a different color. Similarly when the a post it dislikes is supported by another node, all the node's links to Root are tagged with the different color. If it also would have supported that post it assigns them the same color as itself.

Note that are judgments are local! In this basic scheme a user does not need to trust other people's judgments concerning the color of other nodes. Finally most other nodes will have attached to them some judgments according to whether they supported or not posts that the user likes or dislikes. Whether a user likes or dislikes another node is decided on the basis of whether the other node mostly supported or objected to posts that the concerned user would also support or object to.

We ran 1000 experiments with 1000 nodes, and 1000 judged posts. The objective as described above, was to classify users correctly as red and blue according to what posts they objected and took responsibility for. The key results are presented in the left of figure 1: only about 10% of users were incorrectly labeled as red or blue, out of 1000. The full distribution of the number of mis-labeled nodes is plotted. This is rather encouraging, since it shows that even the most simple minded labeling system (majority vote on whether preferences match or not) is capable of providing good levels of filtering, in the absence of sophisticated adversarial behavior.

We do not simply assume that users will not willingly introduce nodes of a different color, but we also attempt to provide the correct set of incentives for this to happen. Figure 1 (right) shows how many nodes got mis-labeled according to whether they introduced a node of a different color. It is clear from the bar plots that those that did introduce a node with different preferences were much more likely to be mis-labeled themselves! This should make users think carefully about who they introduce, and the impact this action may have on their posts being filtered. Note that our simple minded filtering algorithm does not filter on the whole path, as it should, so being introduced by a different node does not come at the same cost – more sophisticated filtering strategies should take this into account.

Finally figures 2 show the bias the the Root color imposes. It is clear that since the Root node is most likely to introduce other nodes of the same color, the nodes that have different preferences from Root will be closer to the leafs of the introduction tree, rather than the center (which is dominated by Root's friends). This results in many more nodes being misclassified as not having Root's color, than being by mistake being classified as Root's color. This is a relatively useful property: if Root and her friends set up an online space, they would wish to set its tone, and care much more about spam and abuse getting through to it, than some good posters being mis-classified. If on the other hand the policy is to never exclude good posts (as it is for email spam filtering) the filtering mechanisms should take this bias into account.

## 7 Distributing functionality

We have been assuming throughout this work that the 'introduction graph' is a tree with user Root at its root. Both of these assumption can be relaxed leading to more flexible, and re-usable abuse resistant publishing systems.

First a single user Alice can be connected to the 'introduction graph' at many points. The simplest extension to our scheme is to allow the same user to be connected under many distinct pseudonyms, and post under any of them. Given our security assumption this assumes that the user has spent the time necessary to convince independent connected users to connect her. A slightly different approach would be for Alice to connect to different points of the introduction graph under the same identity, i.e. public key. This may allow for 'migration', when Alice realizes that a user she is connected to is misbehaving or is blaming her for abuse. This scheme slightly complicates the routing of messages in our ElGamal based construction, since users connected to Alice have multiple paths to root. One could envisage systems with network or source routing of these messages. Source routing might be more difficult since it assumes that users know the full topology of the introduction graph.

It might not be feasible for each different web site to attempt to maintain its own 'introduction graph', since the overhead of introducing users might be considerable. Therefore, there will most probably be a need to share introduction graphs amongst different sites, potentially not trusting each other. A simple extension would be to allow many Roots, where messages are ultimately sent. Each Root performs the protocols as before, considering itself as the main Root of the 'introduction graph'. Network routing would simply 'bubble up' messages towards Root.

Multiple Roots, or just destinations, introduce some interesting problems. Routing of messages, in our interactive Diffie-Hellman based construction, needs to be source routed to ultimately end up at the required destination. It should be possible to modify a traditional mix packet design [11, 13, 14] to provide this service. There is still a need for posters to know the paths to all the destinations they might sent posts to. From a trust point of view it is not clear that users would be comfortable signing up other users for all sites on which they have the ability to post. Yet it is not clear how they could restrict the introduced users' capabilities without knowing where they are posting. The more control introducers want, the fewer the benefits of compounding introduction graphs together – distinct systems start becoming more attractive.

Finally filtering policies might be better implemented by end users rather than enforced centrally by whichever Root. After all, each user is the expert when it comes to their own preferences. Each post in our scheme can be tagged with the objecter and someone who took responsibility for it. Both tags establish a full path to Root, and can be used by users to implement locally filtering policies. Things change slightly if different Roots are present, since some of them may be trusted while others may not (after all abusers themselves could be creating sites, or even users with drastically different preferences – it would be an advantage if our scheme could support differing views in a unified 'introduction graph'). In this case it might be beneficial to transform all paths to be relative to the user. This can be done by appending the paths that connect the user to the different Roots to any paths that may start there. Then one can trivially simplify the paths, in case of a tree strucured 'introduction graph' (e.g. Alice is connected to a Root via path $[\text{Root}, C, B, A]$ and someone with path $[D, E, F, \text{Root}]$ takes responsibility for a post – the path, as far as Alice is concerned becomes $[D, E, F, \text{Root}, C, B, A]$). Since all routes are now relative to Alice she can apply a unified set of filtering rules.

Each user specifying filtering rules on their own maximizes autonomy, yet it also duplicates work when many users mostly agree about what constitutes abuse. Pooling filtering rules themselves, such as blacklists of subtrees, can benefit from the 'introduction graph' being used in the process. Filtering rules can be associated with the full path of their creator, and this can be used as a guide as to whether different sets are in fact likely to be originating from the same creator or their sybils.

## 8 Future work

The study of filtering criteria that minimize the utility of an attacker given a budget for acquiring friends in the 'introduction graph' is left for further study. Different strategies would blacklist different users or branches according to some thresholds of abuse generated. If known, these rules could be used by an adversary to post the maximal amount of abuse before the controlled link into the in-

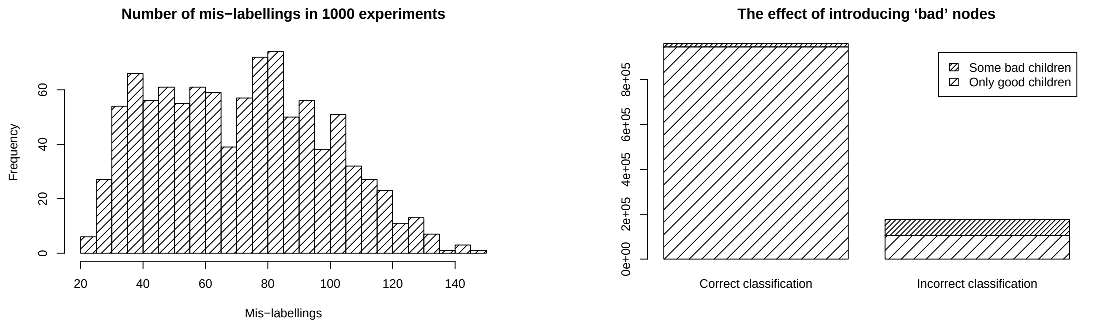


**Figure 1. Histogram of the number of mis-labellings for 1000 users and 1000 posts (left). The effect of introducing nodes of a different color – it leands to a considerably higher chance of being mis-labeled by others (right).**

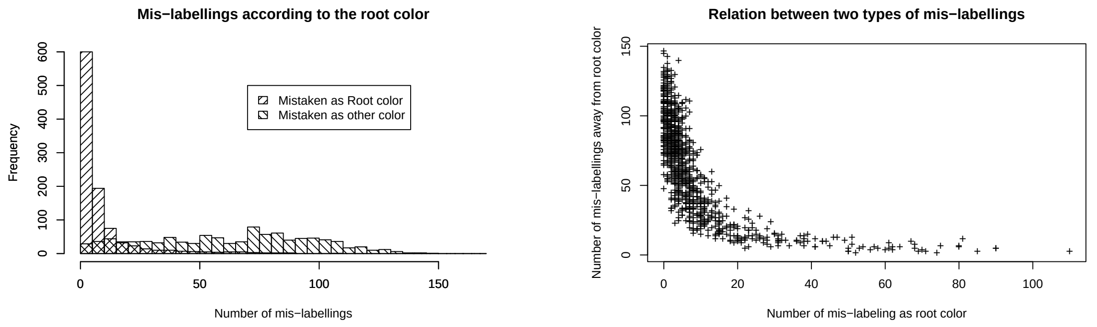


**Figure 2. Mis-labellings according to the Root color.**

troduction graph becomes 'tainted' and useless. So optimal attack strategies and filtering defenses are left for future work.

Our construction is very expensive in terms of communication and computation costs, since each post must involve the full chain of users linking the poster to the Root. Special, delegable or transferable, yet still unlinkable, credential mechanisms could be used to reduce these costs. A poster would then only have to register once, possibly involving their full path, and then should be able to post without any further interactions (until there is an objection lodged). Stengthening the protocols against active and malicious users who aim to identify posters before any objections is also left as future work.

In this work we have concentrated heavily on 'introduction graphs' that are structured as trees, yet it would be conceivable to use any directed graph, with multiple destinations. The cryptographic protocols should then be modified to accommodate routing information, and also to allow efficiently finding routes.

Finally we present in Appendix A an centralized functionality that could be refined down to an ideal functionality to be used to prove the correctness of our construction under the UC model [10] or the Reactive Systems' model [31]. Providing these proofs for our construction or a non-interactive construction is beyond the scope of this paper and its authors.

## 9 Conclusions

We propose a scheme that extracts fuzzy identification information from a social network of posters. Although the process can be made extremely distributed and does not rely on strong trust assumption we argue it can be used to filter persistent abusers from anonymous on-line forums.

Our solution should also be seen as a further step in the tradition of Advogato [28] and Sybil resistant DHTs [15]), finding ways of protecting peer-to-peer systems, or generally protecting systems that cannot rely on strong identification infrastructures against the Sybil attack. Making use of a distributed introduction system, where local trust links and information can be used to reduce the impact of anti-social behavior such as abuse and denial of service seems like a hopeful avenue for further investigation. We can achieve this without ultimately risking the anonymity of any participant.

We argue that the assumption of the emergence or pre-existence of a social network to form such an introduction graph is realistic and can further be cultivated by structuring the incentives of all participants correctly. Identity is then not dependent on 'who you are', an ill-defined proposition, but instead on 'who you know' – a thesis that is in agreement with established wisdom in contemporary sociology [41].

## 10 Acknowledgments

George Danezis is supported by the FWO (Fund for Scientific Research - Flanders) and the EU AEOLUS project.

## 11 References

[1] Independent media center. On-line at `http://www.indymedia.org/`.

[2] Wikipedia, the free encyclopedia. On-line at `http://en.wikipedia.org/wiki/Main_Page`.

[3] Google groups (beta). Website: `http://groups.google.com/`, November 2005.

[4] Openid: an actually distributed identity system. On-line at `http://openid.net/`, July 2005.

[5] M. Arnison. Open publishing is the same as free software. On-line at `http://www.cat.org.au/maffew/cat/openpub.html`, March 2001.

[6] H. S. Baird, A. L. Coates, and R. J. Fateman. Pessimalprint: a reverse turing test. *IJDAR*, 5(2-3):158–163, 2003.

[7] N. Borisov, I. Goldberg, and E. Brewer. Off-the-record communications, or, why not to use PGP. In *Workshop on Privacy in Electronic Society*, 2004.

[8] C. Buchholz. Liberty alliance project - gemeinschaftliche identitätsverwaltung. *Datenschutz und Datensicherheit*, 7(9), 2003.

[9] J. Camenisch and M. Stadler. Proof systems for general statements about discrete logarithms. Technical Report TR 260, 1997.

[10] R. Canetti. Security and composition of multiparty cryptographic protocols. *J. Cryptology*, 13(1):143–202, 2000.

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

[12] G. Danezis and R. Anderson. The economics of resisting censorship. *IEEE Security and Privacy*, 3(1):45–50, 2005.

[13] G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, May 2003.

[14] G. Danezis and B. Laurie. Minx: A simple and efficient anonymous packet format. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2004)*, Washington, DC, USA, October 2004.

[15] G. Danezis, C. Lesniewski-Laas, M. F. Kaashoek, and R. Anderson. Sybil-resistant DHT routing. In di Vimercati et al. [16], pages 305–318.

[16] S. D. C. di Vimercati, P. F. Syverson, and D. Gollmann, editors. *Computer Security - ESORICS 2005, 10th European Symposium on Research in Computer Security, Milan, Italy, September 12-14, 2005, Proceedings*, volume 3679 of *Lecture Notes in Computer Science*. Springer, 2005.

[17] T. Dierks and C. Allen. The TLS protocol. Request for Comments 2246, Network Working Group, January 1999.

[18] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[19] R. Dingledine, N. Mathewson, and P. F. Syverson. Tor: The second-generation onion router. In *USENIX Security Symposium*, pages 303–320. USENIX, 2004.

[20] J. R. Douceur. The sybil attack. In P. Druschel, M. F. Kaashoek, and A. I. T. Rowstron, editors, *IPTPS*, volume 2429 of *Lecture Notes in Computer Science*, pages 251–260. Springer, 2002.

[21] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol. Request for Comments 2616, Network Working Group, June 1999.

[22] F. C. Freiling, T. Holz, and G. Wicherski. Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks. In di Vimercati et al. [16], pages 319–335.

[23] T. E. Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, pages 10–18, 1984.

[24] GMail. On-line at `http://gmail.google.com`.

[25] P. Golle, M. Jakobsson, A. Juels, and P. F. Syverson. Universal re-encryption for mixnets. In T. Okamoto, editor, *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 163–178. Springer, 2004.

[26] B. Kantor and P. Lapsley. Network news transfer protocol. Request for Comments 977, Network Working Group, February 1986.

[27] J. M. Kleinberg. The small-world phenomenon: an algorithmic perspective. In *STOC*, pages 163–170, 2000.

[28] R. L. Levien. *Attack resistant trust metrics*. PhD thesis, University of California at Berkeley, 1995. Draft Thesis.

[29] Livejournal. Wikipedia, the free encyclopedia at `http://en.wikipedia.org/wiki/LiveJournal#Invite_system`, January 2005.

[30] Orkut. On-line at `http://orkut.com`.

[31] B. Pfitzmann, M. Schunter, and M. Waidner. Cryptographic security of reactive systems. *Electr. Notes Theor. Comput. Sci.*, 32, 2000.

[32] M. D. Raimondo, R. Gennaro, and H. Krawczyk. Secure off-the-record messaging. In V. Atluri, S. D. C. di Vimercati, and R. Dingledine, editors, *WPES*, pages 81–89. ACM, 2005.

[33] M. Roe. *Cryptography and Evidence*. PhD thesis, University of Cambridge, Computer Laboratory, 1997.

[34] Slashdot: News for nerds, stuff that matters. On-line at `http://slashdot.org`.

[35] D. X. Song, J. D. Tygar, and D. Zuckerman. Expander graphs for digital stream authentication and robust overlay networks. In *IEEE Symposium on Security and Privacy*, pages 258–, 2002.

[36] F. I. P. Standards. Digital signature standard (dss). Technical Report 186, FIPS, May 19 1994.

[37] N. H. D. Team. Common gateway interface. Technical report, University of Illinois at Urbana - Champaign, 1998. On-line at `http://hoohoo.ncsa.uiuc.edu/cgi/`.

[38] J. Travers and S. Milgram. An experimental study of the small world problem. *Sociometry*, 32(425), 1969.

[39] Internet troll. From Wikipedia, the free encyclopedia. `http://en.wikipedia.org/wiki/Internet_troll`, 29 October 2005.

[40] L. von Ahn, M. Blum, N. J. Hopper, and J. Langford. Captcha: Using hard ai problems for security. In E. Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 294–311. Springer, 2003.

[41] S. Wasserman, K. Faust, D. Iacobucci, and M. Granovetter. *Social Network Analysis : Methods and Applications (Struc-*

*tural Analysis in the Social Sciences)*. Cambridge University Press, Cambridge, UK, 1st edition, 1994.

[42] R. N. Wright, P. Lincoln, and J. K. Millen. Depender graphs: A method of fault-tolerant certificate distribution. *Journal of Computer Security*, 9(4):323–338, 2001.

## A A centralized implementation

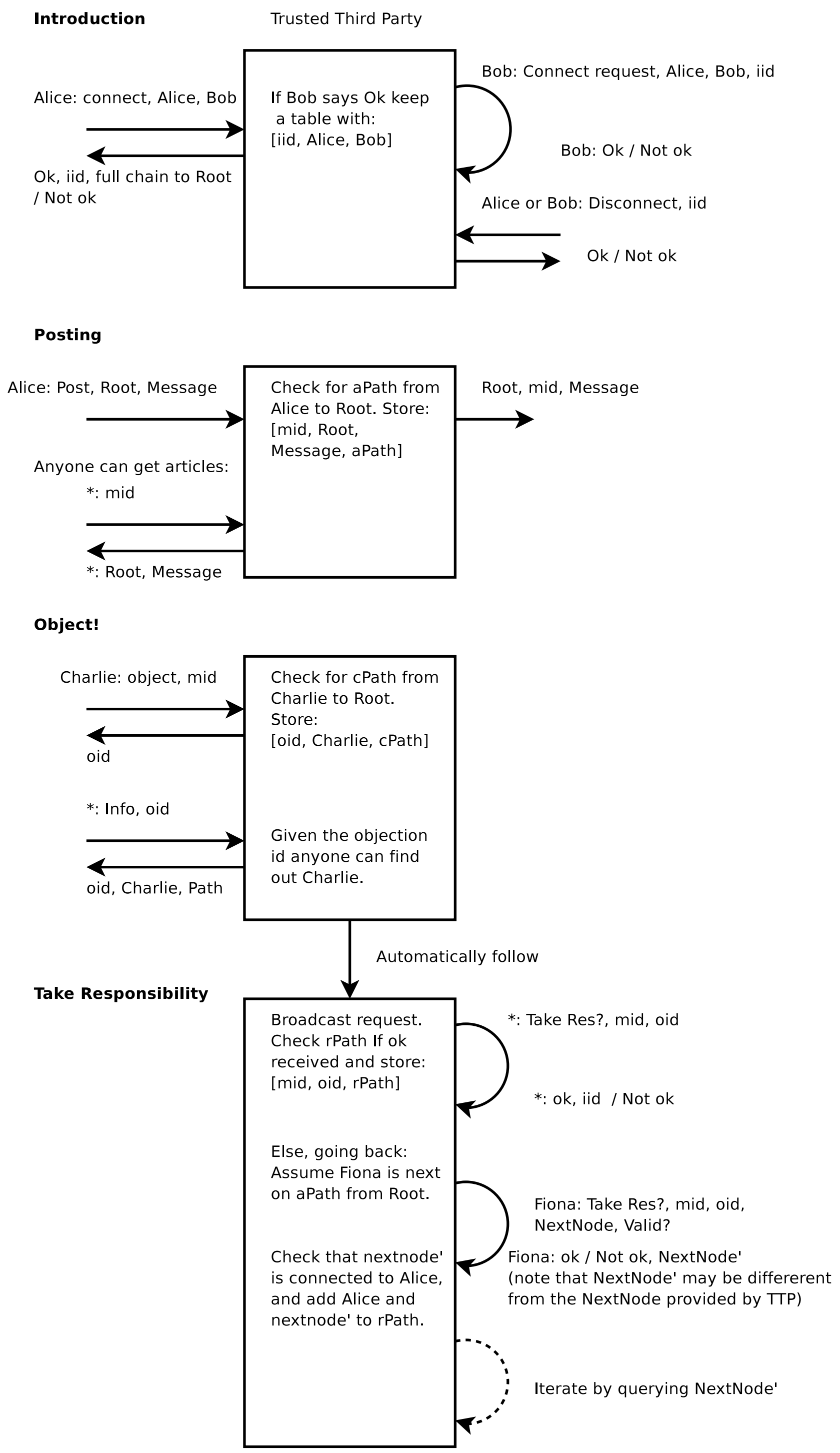
We sketch here how we would implement our abuse resistant publishing mechanism using a trusted third party (TTP). This could be refined down to an 'ideal functionality' to be used to prove cryptographic correctness in Canetti's model [10]. In this case one may show that concrete cryptographic realisation of the scheme do not allow for any more attacks than are possible in this ideal model. It is not out of the question that moderators of websites could be considered trustworthy enough to act as the trusted third party, in which case a centralized implementation could be of practical use.

We first describe the state that the TTP holds and the key interactions with other principals. These are all illustrated in figure 3. The TTP keeps track of established connections between users, and assigns them an identification number *iid* (e.g. $[iid, \text{Alice}, \text{Bob}]$). Upon a request for a connection from Alice to Bob, the TTP simply asks Bob for approval. Bob can at any time severe the connection by handing in the *iid*. As a result the row indexed by *iid* is deleted.

The TTP keeps all messages posted in a table, indexed by *mid*, and containing the path to Root of the sender (aPath), and the message (e.g. $[mid, \text{Root}, \text{Message}, \text{aPath}]$). The TTP also forwards posted messages to Root, and provides them to anyone who requests them (note that the aPath is never directly revealed).

Anyone can object to a post by presenting its *mid*. The full path of the objector (cPath) is stored on a table, indexed by *oid* (e.g. $[oid, \text{Charlie}, \text{cPath}]$). Given the objection number *oid*, anyone can inspect this table, which is made public by the TTP.

Finally the TTP maintains a 'responsibility' table for each objected post. The table stores the message and objection identifiers (*mid* and *oid*) and the path of the user having taken responsibility, or of the progress of the 'taking responsibility' protocol (rPath). The path rPath starts out as only storing Root $[mid, oid, \text{rPath} = \{\text{Root}\}]$. The TTP then asks the first user in the path to the sender for the user after them (lets call them Fiona). There is some subtlety in this request. The TTP provides the user, Fiona, with the actual address of the next user. Fiona is of course free to lie, and provide another user that she is connected to. Furthermore the TTP lets Fiona know if someone has previously lied or not (therefore framing her). Note that there is no possibility for Fiona to prove to anyone else that any lying has taken place. In any case the entry is augmented by Fiona's name, and the next node's name (e.g. $[mid, oid, \text{rPath} = \{\text{Root}, \text{Fiona}, \ldots\}]$). The protocol continues recursivelly as the TTP asks the next node, until someone clains final responsibility, or does not wish to trace any further. The contents of this table are made public at the end of the protocol.



Introduction
Trusted Third Party
Alice: connect, Alice, Bob
If Bob says Ok keep a table with: [iid, Alice, Bob]
Bob: Connect request, Alice, Bob, iid
Bob: Ok / Not ok
Ok, iid, full chain to Root / Not ok
Alice or Bob: Disconnect, iid
Ok / Not ok
Posting
Alice: Post, Root, Message
Check for aPath from Alice to Root. Store: [mid, Root, Message, aPath]
Root, mid, Message
Anyone can get articles:
*: mid
*: Root, Message
Object!
Charlie: object, mid
Check for cPath from Charlie to Root. Store: [oid, Charlie, cPath]
oid
*: Info, oid
Given the objection id anyone can find out Charlie.
oid, Charlie, Path
Automatically follow
Take Responsibility
Broadcast request. Check rPath If ok received and store: [mid, oid, rPath]
*: Take Res?, mid, oid
*: ok, iid / Not ok
Else, going back: Assume Fiona is next on aPath from Root.
Fiona: Take Res?, mid, oid, NextNode, Valid?
Check that nextnode' is connected to Alice, and add Alice and nextnode' to rPath.
Fiona: ok / Not ok, NextNode' (note that NextNode' may be differerent from the NextNode provided by TTP)
Iterate by querying NextNode'

**Figure 3. Ideal Functionality for the proposed mechanism**