Is Privacy possible without Anonymity? The case for microblogging services

Panagiotis Papadopoulos*
Brave Software
panpap@brave.com

Antonis Papadogiannakis*
ProtectWise, USA
apapadog@gmail.com

Michalis Polychronakis
Stony Brook University, USA
mikepo@cs.stonybrook.edu

Evangelos P. Markatos
FORTH-ICS, Greece
markatos@ics.forth.gr

ABSTRACT

Traditional approaches to privacy are usually based on top of anonymizing or pseudonymizing systems. For example, users who would like to protect their identity and/or hide their activities while browsing the web frequently use anonymizing systems (e.g., Tor) or services (e.g., VPNs and proxies). Although anonymizing systems are usually effective, recent revelations suggest that anonymization can be compromised and can be used to provide a false sense of security. In this paper we assume a world where anonymization is (practically) not possible. Imagine, for example, a community where the use of anonymizing systems is frowned upon or even forbidden. Is it possible for users to protect their privacy when they can not hide their identity?

In this paper, we focus our question on users interested in following information channels in microblogging services and we show that *it is possible* for users to protect their privacy even if they can not hide their identity. To do so, we propose two obfuscation-based algorithms and quantify their effectiveness. We show that obfuscation can be used in such a way so that attackers can not use this service to increase their *a priori* knowledge on whether a user is interested in a channel or not.

KEYWORDS

Obfuscation, Anonymous Subscription, User Interests

ACM Reference Format:
Panagiotis Papadopoulos, Antonis Papadogiannakis[1], Michalis Polychronakis, and Evangelos P. Markatos. 2019. Is Privacy possible without Anonymity? The case for microblogging services. In *12th European Workshop on Systems Security (EuroSec '19), March 25–28, 2019, Dresden, Germany*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3301417.3312498

1 INTRODUCTION

Over the past few years we have seen an increase in the deployment and use of microblogging services and news aggregators. Indeed, these aggregators have become very popular, because they provide timely information in short (280-characters-long) nuggets that can be quickly scanned, tailored, absorbed, and shared. As a result of this popularity, an increasing number of users choose to receive their news through such microblogging services such as Twitter, Tumblr, Weibo, Tout, and many more.

Although useful, this timely and personalized information delivery by microblogging services may raise significant privacy concerns. For example, if a user follows a channel of a political party, these microblogging services may be able to infer the user's political beliefs. If the user follows a channel dealing with a particular health problem, the microblogging services may be able to infer that the user is interested in this health problem, or, to make matters worse, that the user is suffering from it. As a result, an adversary who has access to the information provided to microblogging services may easily be able to figure out what the users are interested in, based on the types of channels they follow.

To protect their privacy, users may choose to *hide their real identity* by creating and using a *fake account*. Although such a fake account may give a sense of privacy through pseudonynity, previous research suggests that fake accounts may be traced back to the real identities of their users [9, 24]. To make matters worse, third-party ad networks and tracking services [14, 18, 19] can also be used to reveal the real identity of the user. To protect their privacy even further, in addition to using a fake account, sophisticated users may choose to connect to the Internet through an *anonymizing proxy* or an *anonymizing network* such as Tor [4]. Indeed, anonymizing networks have been shown to be very effective at hiding the IP address of their users. Unfortunately, recent attacks have shown that despite the user of an anonymizing networks, users may still be identified through their browser "fingerprint" [6] or through other characteristics of their device, such as their clock drift rate [13] or their battery drain rate [15]. Even if one manages to hide all identifying characteristics of a device, microblogging services may simply reveal the identity of their users by simply requiring them to log in, much like Facebook does today.

In this paper, instead of joining the arms race between anonymizing networks and their adversaries, we take a slightly different point of view by addressing the following question: "*Let us assume a world where anonymity is (practically) not possible. Can we still protect the privacy of users of a microblogging service?*" We believe that the answer to the above question is *yes*: we believe that even if the user is not able to hide her real identity, she is able to hide her real interests. One way to do this is by using obfuscation: that is, she

*Work was conducted when authors were in FORTH-ICS

EuroSec '19, March 25–28, 2019, Dresden, Germany

ACM ISBN 978-1-4503-6274-0/19/03…$15.00
https://doi.org/10.1145/3301417.3312498

will subscribe to more channels than those she is really interested in, making it difficult for an adversary to find out what her real interests are. By subscribing to more channels that she is really interested in, the user adds noise, protecting, essentially, her real interests. At this point we must admit that adding noise, or *obfuscating*, is not a new idea: it has already been used in loyalty cards, in hiding web searches [2], and even in microblogging services [21].

Despite having been studied and deployed in the past, adding noise is easier said than done. Indeed, recent work has shown that previous obfuscation approaches have been *very* ineffective [8]. We believe that two are the main issues behind for this ineffectiveness: (i) **I1: selecting noise**: It is not clear how to select noise. Naive approaches to noise selection (such as adding random channels or issuing random queries) eventually lead to countermeasures which are able to remove such added random noise, and (ii) **I2: correlation attacks**: The longer a user interacts with a system, the more information she reveals. As a result, the system might be able to launch what in web searches is called a *linkage* attack [8], or what we describe later as a correlation attack.

To deal with the first of the issues above (i.e. **selecting noise**) we developed an analytic model which enables us to explore how much noise is needed to hide the fact that the user (say U) is interested in following a given channel (say S). Our analytic model uses an information-theoretic approach: we assume that the adversary has some *a priori* knowledge about how possible it is for user U to follow channel S (say $P_U(S)$). In order to hide that U is interested in S, our model argues that we should add $\lceil 1/P_U(S) \rceil$ noise channels. In addition to describing *how much* noise we need to add, section 3.1 describes an algorithm that computes *what* kind of noise needs to be added. Indeed, adding randomly chosen channels may seem reasonable, but it is not enough. Random noise can be easily filtered out.

Dealing with the second of the issues identified above (i.e. **I2: correlation attacks**) is much tricker. In a correlation attack, the adversary observes what channels the user is following over time (even if the real channels are obfuscated in a bundle of noise), and tries to find which of them are correlated. The intuition of the attacker is that the channels user U is really interested in will probably be correlated, while the channels that were added as noise were probably not correlated. Thus, if the adversary finds correlated channels, then the user is probably interested in their common theme. For example, if the user follows a channel on alcoholics anonymous (along with a bundle of noise channels) and then a channel on rehab clinics (along with another bunch of noise channels), and then a channel on substance abuse (along again with other noise channels), the adversary may safely conclude that U is interested in alcoholism-related issues. This is because these alcoholism-related channels seem highly correlated among themselves, while the noise channels, would *probably* not be so strongly correlated (if at all). To deal with such correlation attacks we develop an analytic approach based on conditional probabilities. The approach models the information available for a correlation attack and later leads to an algorithm that chooses appropriate noise so as to make the correlation attack very difficult, it not completely impossible.

To summarize, our work makes the following main contributions:

(1) We explore the following question: In a world where anonymity is practically not possible, can we protect the privacy of users of microblogging services? i.e. can we efficiently hide their interests?
(2) We propose *SMOKE*: an obfuscation-based approach to preserve the privacy of the users' interests in microblogging services. Our approach aims to hide a sensitive channel the user is interested in inside other channels that act as noise.
(3) We propose an *analytic approach and a subsequent algorithm that computes the optimal amount and the type of noise that should be added*: too little noise leaves the user exposed, while too much noise is deemed unnecessary.
(4) We show that users who follow a sequence of channels may be subject to correlation attacks. We *describe the attack* and propose an *analytical model* that evaluates these correlations based on conditional probabilities. Based on the model, we *propose an algorithm* which adds appropriate noise that is not subject to these correlation attacks.

2 THREAT MODEL

We assume the existence of a service that enables users to receive personalized news about their topics of interest. Such services include news aggregators and RSS readers like Digg, Apple News, Google News or microblogging services like Twitter, Tout, Tumblr, Weibo. In these services the users are able to subscribe themselves to individual channels or domains that distribute content about various topics. In microblogging services these channels can be accounts of physical persons, of entities such as corporations, of news sites, of public figures like actors, of activists, of politicians' offices, and so on.

By subscribing to these channels the users are able to get in their timeline timely information directly delivered from the information sources meeting their interests. On the other hand, the service operator is capable of recording the users' interests by observing which channels each user follows. All information about the users' interests, along with any data aggregated after them is property of the service operator and can be later sold to advertisers [3, 22] for targeted advertising in ad auctions [20] or considered as company's assets in case of a service's future purchase [12]. As a consequence all this information can be used for a variety of purposes, all of which are beyond the control of individual users. To remedy this potential infringement of the users' privacy, we would like to develop mechanisms to enable the users to hide their actual interests while they consume personalized information.

In this work we assume an "honest but curious" microblogging service able to find the user's interests by monitoring the channels she follows and by mining data produced by the user in order to aggregate and/or correlate any information she provides with her actions inside the web service. On the other hand, we assume that it will not try to actively interfere with the process users are employing to protect their privacy or try to gain more information than what a user is willing, or required, to give. We think such a microblogging service is reasonable in practice, since popular services have a reputation they do not want to jeopardize by becoming hostile against their own clients. As a result we assume that the service will not create any fake channels or fake accounts in

order to gain any more information than its real users are willing to disclose.

Finally, it is important to note at this point, that in this work we assume that users act as *consumers* of information and refrain from *actively exposing* their real interests by *posting* information, replying, retweeting, or sharing their interests in any other way. Recall, that we assume a world where anonymity is (practically) impossible. Actively sharing one's interests in such a world leaves very limited, if any at all, options to protect privacy. [1]

3 OUR APPROACH: OBFUSCATION

3.1 Noise selection

When a user wants to follow a channel S, *SMOKE* instructs the user's profile in the microblogging service to follow N other "noise" channels (say S_1, S_2, ..., S_N) as well.

To make sure that we select the right amount and right type of noise we introduce two points of view:

- ***A priori* knowledge:** the adversary has some confidence that U is interested in S, even *before* U follows S. This confidence may be the result of general background information available (e.g. what percentage of people are alcoholics), or even background information available specifically tailored to U (e.g. what percentage of white males between 40 and 50 are alcoholics). Let us denote this confidence (that user U is interested in following channel S) with $P^p_U(S)$.
- ***A posteriori* knowledge:** after U follows channel S (along possibly with some noise channels S_1 to S_N), this new information may produce a new confidence level that U is interested in S: let us denote this new confidence level with $P^a_U(S)$. For example, if U follows only S and no noise channels, then this confidence will obviously become 1. Indeed, if U follows S and only S and does not add any noise, the adversary will be 100% sure that user U is interested in channel S. As another example, if U follows S along with only one noise channel (say S_1), the adversary will know that U is interested either in S or in S_1, and thus the *a posteriori* confidence will be 0.5: $P^a_U(S) = 0.5$. That is, the adversary is 50% confident that user U is interested in channel S, and 50% confident that user U is interested in channel S_1.

Framed in this context of *a priori* and *a posteriori* knowledge our approach is at odds with the approach of the adversary: the adversary would like to increase as much as possible the *a posteriori* confidence that U is interested in S (i.e. the adversary would like to achieve $P^a_U(S) \gg P^p_U(S)$) while we would like to reduce this *a posteriori* confidence, if possible, to the level where the *a posteriori* confidence is no higher than the *a priori* one: (i.e. we would like to achieve $P^a_U(S) = P^p_U(S)$).

3.1.1 How much noise should be added? To make sure that the *a posteriori* confidence is no higher than the *a priori* one, we advocate that we should add $\lceil 1/P^p_U(S) \rceil - 1$ noise channels. In this way, the adversary will see that the user U is interested in following $\lceil 1/P^p_U(S) \rceil$ channels (S included in them), and his confidence that among all those channels U is interested in S will be $P^a_U(s) = 1/\lceil 1/P^p_U(S) \rceil \approx P^p_U(S)$. Thus the *a posteriori* confidence ($P^a_U(s)$) will be almost equal to the *a priori* one ($P^p_U(S)$) and the adversary will have gained little, if any at all, information. For example, if the adversary is about 10% confident that U is interested in channel S, possibly based on *a priori* knowledge that he has (i.e. $P^a_U(S) = 0.1$), our approach will add 9 noise channels and user U will follow a total of 10 channels (= 9 noise channels plus channel S). Seeing that U is following 10 channels, the adversary will still be about 10% confident that user S is a "real" channel (and not a noise channel). Thus, the *a posteriori* confidence of the adversary will still be 10% and will not have increased compared to his *a priori* confidence.

3.1.2 What kind of noise should be added? As explained above, in order for the adversary to gain little, if any at all, information, we must make sure that the confidence that he has that U is interested in S should be the same as the confidence that he has that U is interested in S_1, in S_2, in S_3, etc. Obviously, if the adversary knows that U is not interested at all in, say S_1, adding S_1 as noise will not be effective: the adversary will immediately realize that S_1 is just noise and U is not interested in S_1. Therefore, the adversary will be able to remove at least one noise channel and increase his confidence level about whether user U is interested in channel S. Therefore, user U should be interested in following all noise channels with the same level of interest she is interested in following channel S. That is, the *a priori* probability that U is interested in S should be the same as the *a priori* probability that U is interested in S_1, which should be the same as the *a priori* probability that U is interested in S_2, etc. Or equivalently:

$$P^p_U(S) = P^p_U(S_1) = P^p_U(S_2) = \ldots = P^p_U(S_N) \tag{1}$$

Finding noise channels that satisfy the equation above we force the adversary to a position where he is unable to determine with accuracy whether the user is actually interested in channel S or one of the channels S_1, S_2, ..., S_N.

As equation (1) above shows, in order to effectively hide the fact that user U is interested in channel S, we need to find N other channels $(S_1, S_2, \ldots, S_N)$ in which the user is equally interested. Unfortunately, it is not very likely to find *exactly* N other channels with all probabilities $P^p_U(S)$, $P^p_U(S_1)$, $P^p_U(S_2)$, ..., $P^p_U(S_N)$ *exactly equal* to each other. It is more probable to find N noise channels with probabilities *close* to each other, but *not equal* to each other. In this case, one might be tempted to select the channels with the probabilities as close as possible to $P^p_U(S)$ and use them as noise. To address this issue, one might be tempted to borrow ideas from *k-anonymity* [23] and *k-subscription* [21] and *always* add a *constant* number of noise channels. Although possible, such a solution would not be efficient. Indeed, in some cases k would be too small increasing the adversary's confidence that the user is interested in channel S, and in other cases k might be too large which will add too much unnecessary noise. Therefore a one-size-fits-all solution (i.e. a constant k) is not appropriate.

Therefore, we introduce a new approach to noise called *bucketing*. This approach places all channels in buckets. When user U is interested in following one channel from the bucket, she is requested to follow *all* channels of this bucket. In this aspect, from the

[1] One might argue that we can use an obfuscation-based approach to protect users who post information. Indeed, in such an approach a user would both to channels she follows and to channels she is not interested in. We are afraid that such an aggressive approach would lead to information pollution and should probably not be encouraged.

adversary's point of view, all channels of the same bucket are indistinguishable: if the user is interested in following *any one* of them, she is requested to follow *all* of them. Therefore, the adversary can not distinguish any of the channels in the bucket: since users always follow all of them, no matter which individual channel they are interested in, the adversary can point pinpoint which individual channel a user is actually interested in.

Placing the channels in buckets is not easy, however. One reason for this is that the size of each bucket should not be constant; it should depend on the probability of the channels it contains and more specifically, on the lowest channel probability contained in the bucket. For example, if user U interested in channel S with probability $P_U^p(S) = 10\%$, the bucket should contain around 10 channels. If it contains much less than 10 channels, then the adversary will be able to increase its *a posteriori* confidence (that U is interested in S) well above 10%. On the other hand, if it contains many more than 10 channels, user U will be forced to follow many more noise channels than necessary.

To define the buckets our algorithm proceeds as follows:

- It orders all channels C according to their *a priori* probability $P_U^p(C)$ in decreasing order.
- It constructs the set of channels that have probabilities in the range $[1/2, 1)$. If there are more than two such channels it creates buckets of size two. Each bucket contains at least two channels each of them with probability at least $1/2$.
- It then constructs the set of channels that have probabilities in the range $[1/3, 1/2)$. If there are more than three such channels in this range, it creates buckets of size three. If, however, no three such channels are found, the algorithm will try to find *four* channels in the range $[1/4, 1/2)$. [2] If unsuccessful, it will try to find *five* channels in the range $[1/5, 1/2)$, and, if unsuccessful, it will try to find *six* channels in the range $[1/6, 1/2)$, etc. until it succeeds.

In Algorithm 1 we present the bucket algorithm in pseudocode. At first we define a range of values: $[\frac{1}{k}, \frac{1}{i})$ and we set as cursor the left edge (i.e. $1/k$). Then, in each step we check if there are k channels that their probabilities are in this range (i.e. $[\frac{1}{k}, \frac{1}{i})$ - if not, we increase k and check if there are k channels that their probabilities are in this range (i.e. $[\frac{1}{k}, \frac{1}{i})$ - if not, we increase k again and so on, until we find k channels that their probabilities are in the range $[\frac{1}{k}, \frac{1}{i}$. In this way, the bucket algorithm will create a bucket in the field of values $[1/k, 1/i)$ that contain at least k channels. Let us give another example as well: let's assume that we have channels with probabilities $P(A) = 0.6$, $P(B) = 0.55$, $P(C) = 0.44$, $P(D) = 0.37$, $P(E) = 0.30$, and $P(F) = 0.29$. Our bucket creation algorithm will create one bucket in the range $[1/2, 1)$ which will contain 2 channels (i.e. A and B). It will then try to find 3 channels in the range $[1/3, 1/2)$ but it will not succeed (because only two channels are in this range: channel C and channel D); then, it will try to find four channels in the range $[1/4, 1/2)$, it will succeed and it will create a bucket with them (i.e. C, D, E, and F).

[2]The reason for this choice is the following. Let us assume that there are no three channels in the range [1/3, 1/2]. Let us further assume that there is only one channel in the range [1/3, 1/2]. Let us assume that we create a bucket that contains only this channel. If, at some point in the future, the user would be interested to follow this channel, our system would ask the user to follow all channels of the bucket. Since, however the bucket contains only one channel, the adversary will be 100% sure that the user is interested in this channel . Similarly, if there are only two channels in the range [1/3, 1/2], if we create a bucket out of these two channels, and ask the user to follow the bucket when the user is interested in either of the channels, then the attacker will know with probability 50% that the user is interested in the channel. Since the *a priori* probability was less than 1/2 and the *a posteriori* probability is 50%, then the attacker increased his knowledge with regards to whether the user is interested in this channel.

ALGORITHM 1: Bucket creation algorithm

```
#define M 1000 {maximum obfuscation factor}
B = ∅
for (i = 1 to M) do
  for (k = i + 1 to M) do
    B = {set of channels with a priori probability in [1/k, 1/i)} ∪B
    if (|B| >= k) then
      while (|B| >= k) do
        L = {set of first k channels from B}
        newbucket = L
        B = B \ L
      end while
      i = k
      break
      {break the inner k loop and continue with the i loop}
    end if
  end for
end for
```

After creating the buckets, *SMOKE* proceeds as follows:

> If user U is interested in following channel S which belongs in bucket B, then U is requested to follow all channels of bucket B.

Theorem 3.1. *When user U is interested in following channel S, Algorithm 1 places S in such a bucket so that the* a posteriori *probability of U being interested in S is not higher than the* a priori *probability of U being interested in S.*

Proof. Let us assume that user U is interested in following channel S. Let us also assume that the adversary knows the *a priori* probability $P_U^p(S)$. Let us also assume that there exists an integer l so that $1/(l+1) < P_U^p(S) \leq 1/l$. Then Algorithm 1 places channel S in a bucket with $k-1$ channels whose *a priori* probabilities are all between $1/k$ and $1/l$. That is,

$$1/k < P_U^p(S_i) \leq 1/l \tag{2}$$

As we said, the *a priori* probability that U is interested in S is $P_U^p(S)$. The *a posteriori* probability that U is interested in S is $P_U^a(S) = \frac{P_U^p(S)}{\sum_{i=1}^k P_U^p(S_i)}$. We need to show that this is less or equal to the *a priori* probability or equivalently $\frac{P_U^p(S)}{\sum_{i=1}^k P_U^p(S_i)} \leq P_U^p(S)$ or

$$\sum_{i=1}^{k} P_U^p(S_i) \geq 1 \tag{3}$$

But from equation 2 we know that $\forall\, i, P_U^p(S_i) > 1/k$. Plugging this into inequality 3 we get $\sum_{i=1}^k P_U^p(S_i) \geq \sum_{i=1}^k 1/k = 1$ and thus $P_U^a(S) = \frac{P_U^p(S)}{\sum_{i=1}^k P_U^p(S_i)} \leq P_U^p(S)$

□

3.1.3 Bounds in the error estimation. The described approach (i.e. Algorithm 1) is based on the fact that for each user U and each channel S we know the *a priori* probability that U is interested in S: $P^p_U(S)$. One might reasonably argue that this probability is not known, or is not accurately estimated. In this section we acknowledge this fact and we take it one step further to focus on the following question: *if we have only an estimate of $P^p_U(S)$, how is it expected to influence our results*?

Let us illustrate this with an example. Suppose that we have a user U who would like to follow the channel S of a political party. Suppose that the general feeling out there is that this party has about 10% of the voters, and so a general approximation of $P^p_U(S)$ would be about 0.10. Our approach, which does not have inside information would reasonably assume that $P^p_U(S) = 0.10$. Assume, however, that the microblogging service has more information and knows that people who fit the profile of U are two times less likely to vote for this party and thus the correct estimate of $P^p_U(S)$ would be 0.05, and not 0.10.

Since our approach assumes that the probability is 0.1, it would probably choose 9 more channels as noise and thus it would present the microblogging service with a request to follow 10 channels (one of the being S). Since the microblogging service was presented with a request for 10 channels, it would assume that the *a posteriori* probability would be $P^a_U(S) = 0.1$, or twice as much as the *a priori* probability.

On general it can be shown that the following theorem is true:

Theorem 3.2. *If our estimate for the* a priori *probability $P^p_U(S)$ is off by a factor of $f(> 1)$ compared to the adversary's estimate, then the adversary's estimate for the* a posteriori *probability $P^a_U(S)$ will increase at most by a factor of f.*

Proof. We will assume two cases: Case 1: we underestimate the *a priori* probability, and Case 2: we overestimate the *a priori* probability.

In Case 1 where we underestimate the *a priori* probability, let us say that our estimate is $P^p_U(S)$ and that the adversary's estimate is $f \times P^p_U(S)$. Based on our approach we will select [3] $\lceil 1/(P^p_U(S)) \rceil$ channels to follow and thus based on the number of channels to follow the adversary's *a posteriori* probability would be $P^p_U(S)$ which is much lower than the adversary's *a priori* probability $f \times P^p_U(S)$. In this case, the adversary will just ignore the fact that user U is following S among other noise channels and stick to his *a priori* probability.

In Case 2 where we overestimate the *a priori* probability, let us say that our estimate is $P^p_U(S)$ and that the adversary's estimate is $P^p_U(S)/f$. Based on our approach we will select $\lceil 1/(P^p_U(S)) \rceil$ channels to follow and thus based on the number of channels to follow the adversary's *a posteriori* probability would be $P^p_U(S)$ which is higher than the adversary's *a priori* probability $P^p_U(S)/f$ by a factor of f. Thus, the adversary has increased his confidence by a factor of f.

To summarize, in Case 1 the adversary's estimate for the *a posteriori* probability $P^a_U(S)$ has not increased and in Case 2 it has increased by a factor of f. □

The above theorem gives us a very elegant results that bounds the error in the estimation of of the *a priori* probability:

> If the *a priori* is overestimated by a factor of f this will increase the adversary's belief by no more than a factor of f. If the *a priori* is underestimated by a factor of f, this will not increase the adversary's belief.

In addition, theorem provides a nice way for us to make conservative choices. That is, if we are concerned that we have overestimated a probability by a factor of f, we may just decrease it by the same factor and the adversary will have gained no knowledge at all.

3.2 Correlation Attacks

As explained in section 1 the longer a user interacts with a system, the more information she reveals. This information can be potentially used by an adversary to filter out some noise channels and find the real interests of the users. This is what we call a *correlation* attack. Let us use an example to illustrate the attack. Let us assume that user U is following channel S (along with a bunch of noise channels). Let us also assume that U is interested in following T as well. Let us also assume that T publishes content semantically correlated with the content of S. [4] Such cases, constitute a privacy risk since the microblogging service is able to identify this correlation and as a consequence the fact that U interested in sensitive channel S and not in any of the noise channels. To summarize, when a user follows a channel new channel T, there are two cases:

- T is *semantically unrelated* to the previously followed channel S, or
- T is *semantically related* to S,

In the former case the fact that user follows T reveals no more information about whether the user is interested in S. In the latter case, however, the fact that the user is following T (which is semantically related to S) gives some more confidence that the user is really interested in S and not in the rest of the noise channels followed.

Let's try, at this point, to mathematically define the notion of what is called *semantically related* channels. Assuming $P(S)$ is the probability that the user is interested in channel S, then $P(S|T)$ is the probability that she is actually interested in a already followed sensitive channel S given that she now follows channel T. Thus, channels S and T are semantically related if $P(S|T) \neq P(S)$. So, similar to the single channel noise selection, in this case we select noise following the equation:

$$P(S|T) = P(S_1|T_1) = P(S_2|T_2) = \ldots = P(S_N - 1|T_N - 1) \tag{4}$$

where, S_1, S_2,... $S_N - 1$ are the already followed noise channels for sensitive channel S. From the formula above, we can see that each couple of channels: S_1 and T_1, S_2 and T_2,..., $S_N - 1$ and $T_N - 1$ are also correlated channels. Hence, we select the appropriate noise channels for T based on the already followed noise channels

[3] Without loss of generality we assume that there exist $\lceil 1/(P^p_U(S)) \rceil$ channels with *a priori* probability $(P^p_U(S))$ each. If no such channels can be found, the proof is similar in concept.

[4] For example, S can be an information channel on cancer and T can be a support channel for cancer survivors.

of S creating this way *noise correlations*. As a result, we hide the correlated couple of channels that the user is interested in, inside other correlated channel pairs. In the same manner, in case of more than two correlated channels we use the same algorithm to introduce noise channels formed as couples.

4 RELATED WORK

Papadopoulos et al. in [21] propose to hide a user's subscription from the microblogging service by selecting $k-1$ other noise channels in which the channel that the user is actual interested in will be hidden, and afterwards by encouraging the user to subscribe to all these k channels. They also propose two algorithms, the uniform and the proportional algorithm, for random selection of the noise channels from a set with sensitive channels that is commons among the users of the microblogging service. Although our approach is similar to k-subscription, k-subscription deals only with unrelated channels and can not take into account correlations between different channels.

There are similar approaches in different context, aiming to hide user queries in search engines. Howe and Nissenbaum in [11] proposed *TrackMeNot*, a system designed to hide a user's real interest from a search engine. More specifically, for each real query submitted to the search engine, by a user, the TrackMeNot browser plugin [10] also submits several other queries to confuse the search engine and introduce doubt for the user's real queries. TrackMeNot has been proven to be seminal work in the field and has led to the development of several other methods. For example, GooPIR [5], as an extend of TrackMeNot, proposes an approach that is robust against timing attacks. For each real query, the user wants to submit, GooPIR constructs $k-1$ other queries and submits all k of them at the same time to the search engine. As a consequence, the search engine cannot construct a timing model on the user's real queries.

Murugesan and Clifton in [16] propose *Plausibly Deniable Search* (PDS) aiming to provide plausible deniability to users, with respect to their search queries. Similar to GooPIR mentioned above, each real query is accompanied by $k-1$ other noise queries. In addition, each real query is also brought into a *canonical from* preventing this way identifiability based on typos and/or grammar/syntax of the queries [1, 17]. In the same spirit, Ye et al. [25] propose noise injection for search privacy protection. They give a lower bound for the amount of noise queries required for perfect privacy protection and provide the optimal protection given the number of noise queries. Although the above systems are very effective at hiding *one* real query in a crowd of k queries, a determined adversary may be able to find a user's interests by studying successive sequences of queries. Indeed, if a user consistently generates authentic queries on a particular topic, but the $k-1$ "noise" queries added are on several different topics, then the adversary may easily find the user's real interests using clustering approaches. To protect against clustering attacks, PRAW [7] generates dummy queries on topics related to the topics the user is interested in.

5 CONCLUSION

In this paper we explored whether it is possible for users to hide their interests in microblogging services in a setting where anonymity is not possible. We showed that *it is possible* for users to protect their privacy even if they can not hide their identity. To do so, we proposed two obfuscation-based algorithms and quantified their effectiveness. We believe that this result is very encouraging and that it will probably encourage more privacy research in this setting.

Acknowledgements: The research leading to these results has received funding from European Union's Marie Sklodowska-Curie grant agreement 690972 (PROTASIS). The paper reflects only the authors' view and the Agency and the Commission are not responsible for any use that may be made of the information it contains.

REFERENCES

[1] S. Afroz, M. Brennan, and R. Greenstadt. Detecting Hoaxes, Frauds, and Deception in Writing Style Online. In , pages 461–475, Washington, DC, USA, 2012. IEEE Computer Society.

[2] E. Balsa, C. Troncoso, and C. Diaz. Ob-pws: Obfuscation-based private web search. In *SP '12*, 2012.

[3] F. T. Commission. F.t.c. sues failed website, toysmart.com, for deceptively offering for sale personal information of website visitors. http://www.ftc.gov/news-events/press-releases/2000/07/ftc-sues-failed-website-toysmartcom-deceptively-offering-sale, 2000.

[4] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *SSYM'04*.

[5] Domingo-Ferrer, Josep and Solanas, Agusti and Castellà-Roca, Jordi. h(k)-private information retrieval from privacy-uncooperative queryable databases. *Online Information Review*, 33(4):720–744, 2009.

[6] P. Eckersley. How unique is your web browser? In , 2010.

[7] Y. Elovici, C. Glezer, and B. Shapira. Enhancing Customer Privacy While Searching for Products and Services on the World Wide Web. *Internet Research*, 2005.

[8] A. Gervais, R. Shokri, A. Singla, S. Capkun, and V. Lenders. Quantifying web-search privacy. In *CCS '14*, pages 966–977, New York, NY, USA, 2014. ACM.

[9] O. Goga, D. Perito, H. Lei, R. Teixeira, and R. Sommer. Large-scale correlation of accounts across social networks. Technical report, International Computer Science Institute (ICSI), 2013.

[10] P. Gralla. Trackmenot firefox add-on keeps search engine data profilers confused. http://www.pcworld.com/article/235741/TrackMeNot.html, July 2011.

[11] D. C. Howe and H. Nissenbaum. TrackMeNot: Resisting surveillance in web search. In I. Kerr, V. Steeves, and C. Lucock, editors, *Lessons from the Identity Trail: Anonymity, Privacy, and Identity in a Networked Society*, chapter 23, pages 417–436. Oxford University Press, Oxford, UK, 2009.

[12] L. King. Facebook, whatsapp, and the insatiable appetite for data. http://www.forbes.com/sites/leoking/2014/03/06/facebook-whatsapp-and-the-insatiable-appetite-for-data/, March 2014.

[13] T. Kohno, A. Broido, and K. Claffy. Remote physical device fingerprinting. *Dependable and Secure Computing, IEEE Transactions on*, 2005.

[14] G. Kontaxis, M. Polychronakis, A. D. Keromytis, and E. P. Markatos. Privacy-preserving social plugins. In *Security'12*, 2012.

[15] N. Lomas. Battery attributes can be used to track web users. http://techcrunch.com/2015/08/04/battery-attributes-can-be-used-to-track-web-users/, 2015.

[16] M. Murugesan and C. Clifton. Providing privacy through plausibly deniable search. In *SDM'09*, pages 768–779, 2009.

[17] A. Narayanan, H. Paskov, N. Z. Gong, J. Bethencourt, E. Stefanov, E. C. R. Shin, and D. Song. On the feasibility of internet-scale author identification. In *SP '12*, pages 300–314, Washington, DC, USA, 2012. IEEE Computer Society.

[18] P. Papadopoulos, N. Kourtellis, and E. P. Markatos. Exclusive: How the (synced) cookie monster breached my encrypted vpn session. In *EuroSec'18*.

[19] P. Papadopoulos, N. Kourtellis, and E. P. Markatos. Cookie synchronization: Everything you always wanted to know but were afraid to ask. *CoRR*, abs/1805.10505, 2018.

[20] P. Papadopoulos, N. Kourtellis, P. R. Rodriguez, and N. Laoutaris. If you are not paying for it, you are the product: How much do advertisers pay to reach you? In *IMC'17*.

[21] P. Papadopoulos, A. Papadogiannakis, M. Polychronakis, A. Zarras, T. Holz, and E. P. Markatos. K-subscription: Privacy-preserving microblogging browsing through obfuscation. In *ACSAC '13*.

[22] RT. Privacy betrayed: Twitter sells multi-billion tweet archive. http://www.rt.com/news/twitter-sells-tweet-archive-529/, February 2012.

[23] L. Sweeney. K-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, Oct. 2002.

[24] M. N. Szomszor, I. Cantador, and H. Alani. Correlating user profiles from multiple folksonomies. In *HT '08*.

[25] S. Ye, F. Wu, R. Pandey, and H. Chen. Noise injection for search privacy protection. In *CSE '09*, 2009.