

# On the Privacy Impacts of Publicly Leaked Password Databases

Olivier Heen, Christoph Neumann

▶ **To cite this version:**

Olivier Heen, Christoph Neumann. On the Privacy Impacts of Publicly Leaked Password Databases. 14th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2017), Jul 2017, Bonn, Germany. 10.1007/978-3-319-60876-1_16 . hal-01624534

**HAL Id: hal-01624534**
**https://hal.science/hal-01624534**

Submitted on 26 Oct 2017

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# On the privacy impacts of publicly leaked password databases

Olivier Heen and Christoph Neumann

Technicolor, Rennes, France
{olivier.heen, christoph.neumann}@technicolor.com

**Abstract.** Regularly, hackers steal data sets containing user identifiers and passwords. Often these data sets become publicly available. The most prominent and important leaks use bad password protection mechanisms, e.g. rely on unsalted password hashes, despite longtime known recommendations. The accumulation of leaked password data sets allows the research community to study the problems of password strength estimation, password breaking and to conduct usability and usage studies. The impact of these leaks in terms of privacy has not been studied.
In this paper, we consider attackers trying to break the privacy of users, while not breaking a single password. We consider attacks revealing that distinct identifiers are in fact used by the same physical person. We evaluate large scale *linkability* attacks based on properties and relations between identifiers and password information. With these attacks, stronger passwords lead to better predictions. Using a leaked and publicly available data set containing $130 \times 10^6$ encrypted passwords, we show that a privacy attacker is able to build a database containing the multiple identifiers of people, including their secret identifiers. We illustrate potential consequences by showing that a privacy attacker is capable of deanonymizing (potentially embarrassing) secret identifiers by intersecting several leaked password databases.

## 1 Introduction

Data sets containing user identifiers and password related information are regularly published. In general, these data sets have been hijacked by some hackers, who then published the data on the Internet. The list of such leaks is quite long and only a small fraction of it is listed in Table 1. Taken all together this constitutes a large corpus of personal information. Two factors are worrying in this context. First, the size of the leaks tends to increase, putting more and more users at risk. Second, the passwords are often insufficiently protected, despite long time known recommendations.[1] The most prominent and important leaks over the last years - some of which are listed in Table 1 - use bad password protection mechanisms.

It is commonly accepted that insufficiently protected passwords - e.g. relying on unsalted password hashes or using the same encryption key - have weak

[1] Such as recalled in the OWASP Password Storage Cheat Sheet

**Table 1.** Large leaked password databases. Most of them use password-equivalents.

Top 5 confirmed password leaks on "';--have i been pwned?" [a] in February 2017:

| Site | #identifiers | Year[b] | Protection | Password-equivalent? |
|---|---|---|---|---|
| MySpace | 360 million | 2008 (2016) | hash, sha1 | yes |
| LinkedIn | 164 million | 2012 (2016) | hash, sha1 | yes |
| Adobe | 153 million | 2013 | encryption, 3des | yes |
| VK | 100 million | 2012 (2016) | plaintext | yes |
| Rambler | 100 million | 2014 (2016) | plaintext | yes |

Data sets used in this paper:

| Name | #identifiers | Category | Protection | Password-equivalent? |
|---|---|---|---|---|
| $A$ | $1,5$ million | adult | plaintext | yes |
| $B$ | 1 million | social network | salt + hash, md5 | no |
| $C$ | 164 million | social network | hash, sha1 | yes |
| $D$ | 153 million | software company | encryption, 3des | yes |

[a] https://haveibeenpwned.com/PwnedWebsites - retrieved February 2017.

[b] If the release year is different from the hack year, the release year is provided in parenthesis.

security properties and ease the breaking of passwords. Still, in the light of the important number of leaks that actually use a bad password protection mechanism, it is important to understand all the different types of attacks - including privacy attacks - that an attacker can perform.

These last few years, research focused on password user studies [5, 25, 1], password breaking [26, 18] and estimation of password strength [6, 15, 24, 1, 2, 13, 3]. Most existing attacks apply to passwords that are used by an important number of users. E.g. dictionary attacks or grammar based attacks [26] focus on passwords that a human would generate. Password popularity is also used to measure the strength of a password [24, 7, 14]. The intuition is that the more frequent a password is, the less secure it is. Conversely, rare password are found to be more secure. The popularity distribution of passwords typically follows a Zipf law [14, 7] - meaning that the frequency of a password is inversely proportional to its rank - as exemplified for the data set $D$ used in our study in Figure 1. Related work mainly concentrates on frequent passwords represented on the left hand side of this figure.

In this work, we focus on rare passwords (i.e. supposedly secure passwords), corresponding to the heavy tail of the password distribution. In our example distribution (Figure 1), this corresponds to the passwords located at the bottom right of the curve. We worry about the information that a privacy attacker can find automatically *without* recovering the password clear text. Data sets with insufficiently protected passwords provide *password-equivalents* that can be reused in subsequent attacks, even though the corresponding clear text password is never disclosed. Typical password-equivalents are unsalted password hashes and passwords encrypted with a fixed unknown key.

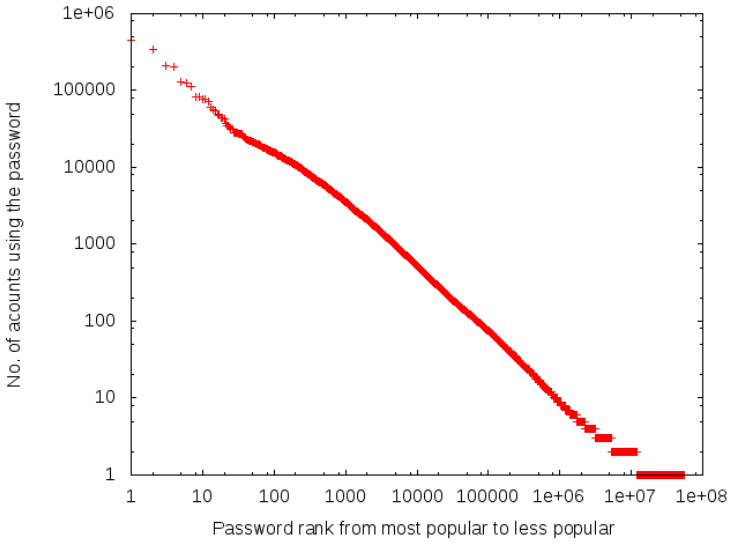



**Fig. 1.** Distribution of passwords in the data set $D$ used in our study. For each password appearing in the data set we compute its rank in the data set (horizontal axis) and its number of occurrences (vertical axis). The relatively flat aspect on a log/log representation is characteristic of a Zipf law [21].

*Contributions* We introduce a model for leaked identifier and password data sets regarding privacy matters. We formalize the notion of *password-equivalents*. We further describe the privacy attacker and define the tools and relations she will operate on identifier names and passwords.

We present classifiers for linking identifiers and revealing secret links, i.e. links that people do not reveal publicly. Using these classifiers, for a subset of these secret links the privacy attacker is able to deanonymize the associated secret identifiers.

We use a publicly leaked data set (named $D$ in this paper) to evaluate our classifiers. It is one of the largest publicly available data set in its kind containing $153 \times 10^6$ identifiers and $130 \times 10^6$ encrypted passwords. With this dataset we show that a privacy attacker can link millions of identifiers, and deanonymize hundreds of thousands secret identifiers. Having no ground truth (for obvious privacy reasons), we estimate the precision of the classifiers through indirect measurements. Finally, we illustrate the consequences of a privacy attack that deanonymizes secret identifiers appearing in a data set related to adult content (denoted $A$ in this paper), by intersecting $A$ with $D$.

## 2 Problem statement

This section defines the problem being addressed by our work. We introduce the attacker model and define linkability properties. We finish with a note on legal and ethical aspects and the precautions used throughout our experiments.

### 2.1 Attacker model

We consider a *privacy attacker* that retrieved a data set $D$ containing identifiers (e.g. *name@mail*) and password-equivalents. The *privacy attacker*'s objective is to link identifiers within $D$. In contrast, most related work consider the *confidentiality attacker* willing to retrieve clear text passwords.

The *privacy attacker* is interested in building a large database revealing sensitive links of potential victims. This is different from an attacker focusing on a specific person and gathering information on this victim (via search engines, online social networks, approaching the victim, etc.). The *privacy attacker* might not carry out the final targeted attack, such as targeted scam or spam, herself. Instead, she might just sell the resulting database.

The privacy attacks presented in this paper target the passwords that are less sensitive to password breaking. Consequently, users that are subject to the privacy attacker are not necessarily subject to the confidentiality attacker and vice-versa.

### 2.2 Model and Definitions

Throughout the paper, we use the privacy related notions defined hereafter.

**Definition 1.** *A **password-equivalent** is the output of a function $f(p)$ applied to a plain-text password $p$. $f(p)$ is a function in its strict sense, meaning that each plain-text password is related to exactly one password-equivalent.*

With this definition a password-equivalent encompasses unsalted hash values such as $sha1(p)$, hash values with a fixed salt such as $sha1(p."0xdeadbeef")$, unsalted encrypted values such as $3DES(p, S)$ where $S$ is a secret key, etc. This excludes outputs of randomized hash-functions as in [11]. In this paper, we consider $f(p)$ to be injective; we are thus neglecting collisions of hashes.

Consistently with [19], we define linkability and $k$-linkability.

**Definition 2.** *Identifiers $x$ and $y$ are **linked**, denoted $L(x, y)$, if $x$ and $y$ are identifiers of the same real person.*

We also introduce the informal notions of *secret link* and *secret identifier*.

$L'(x, y)$ is a *secret link* if the attributes of $x$ provide no information about $y$. Informally, $x$ and $y$ hide their connection, e.g. by using identifier names that are sufficiently different to not reveal the link.

$x$ is a *secret identifier* of $y$ (i) if there exists a secret link $L(x, y)$ and (ii) if the identifier $x$ does not reveal the identity of the person (the identity being e.g. the person's family name or the URL of a public profile page) while the identifier $y$ does.

**Definition 3.** *Given a data set $D$ of identifiers, a person is* ***k-linkable*** *in $D$ if there exists a subset $\hat{D}$ of $D$ such that $L(x_i, x_j); \forall x_i, x_j \in \hat{D}$ and $|\hat{D}| = k$.*

In this work, we evaluate a linkability attack on the data set $D$. This linkability attack infers links between identifiers, and we provide a lower bound probability $p$ that identifiers are indeed linked. More formally, we define $p$ as $Pr[L(x_i, x_j); \forall x_i, x_j \in \hat{D}] \geq p$. In Section 5, we provide estimates and statistics for $k$ and $p$.

The *privacy attacker* employs similarities to compare identifiers. One first similarity, denoted $ls(x, y)$, is the complement of the normalized Levenshtein distance between character strings $x$ and $y$. A second similarity, denoted $jw(x, y)$, is the Jaro-Winkler similarity. The Jaro-Winkler similarity was created for re-conciliating user names from heterogeneous databases, the so-called *record linkage problem*. The Jaro-Winkler similarity provides good results for short strings such as names [4]. Noticeably, $jw(x, y)$ is generally higher than $ls(x, y)$ for pairwise comparisons of strings such as: ‘‘ic”, ‘‘icomputing”, ‘‘ingrid.computing”, ‘‘computing.ingrid”.

Last, the *privacy attacker* computes the sets defined below.

**Definition 4.** *For any identifier $x$ in $D$, let $sp(x) = \{y | y \in D$ and $pwd(y) = pwd(x)\}$, the* ***Same Rare Password*** *function is:*

$$srp_r(x) = \begin{cases} sp(x) \text{ if } |sp(x)| = r \\ \emptyset \text{ otherwise} \end{cases}$$

*The extension of $srp_r$ to subsets of $D$ is $srp_r(\{x_1, \ldots, x_n\}) = \bigcup_{i=1}^{n} srp_r(x_i)$*

In practice, we consider values in the range $2 \leq r \leq 9$.

### 2.3 Note on ethics

Dealing with passwords and personal identifiers raises legal and ethical concerns. Accordingly, we took a set of considerations and employed appropriate precautions.

The objective of this work is to understand, as researchers, the privacy implications of password leaks, poor password storage practices, and to raise awareness amongst colleagues, administrators and the community at large.

As a first precaution, all our results are non-nominative, i.e., they do not include any real personal identifiers. In particular, in this paper, we build examples such that: (i) the exemplified property is still clear, (ii) no single element leads back to any real identifier attribute. The example names, emails and encrypted passwords are invented, such as ‘‘ingrid.computing” in Table 2.

As a second precaution, for all treatments not requiring word distance computations or requiring the detection of some pattern, we anonymize the name part of the account using a keyed SHA256 function. For all treatments requiring word distance computations or requiring the detection of some pattern (e.g. detection of separators) we perform the same anonymization operation just after

```
10...89-|--|-ingrid.computing@comp.com-|-32gt...dfmQhQa...Dzfl==-|-same|--
13...25-|--|-0628...09@mail.uk-|-32gt...dfmQhQa...Dzfl==-|-usual|--
```

| *uid* | *pwdl* | *pwdr* | *name* | *mail* | *hint* |
|---|---|---|---|---|---|
| 10...89 | gt...dfm | Qa...D | ingrid.computing | comp.com | same |
| 13...25 | gt...dfm | Qa...D | 0628...09 | mail.uk | usual |

**Fig. 2.** Top: original text. Bottom: result after normalization. *uid*: internal user identifier zero-padded to 9 digits. *pwdl*: significant bytes of the encrypted left part of the password. *pwdr*: significant bytes of the encrypted right part of the password if any. *name*: identifier before '@' if any. *mail*: identifier after '@'. *hint*: hint string.

the distance computation or pattern detection. These precautions guarantee that no real identity appears as a result of a treatment.

As a third precaution, we key-hashed the passwords regardless whether there were already protected or not in their initial dataset. None of our treatments require the knowledge of the real password.

In addition, we took classical security measures to protect and clean the files and the programs used for this study.

Our results rely on leaked and publicly available password data sets, and there is a debate whether researchers should use such data sets (see [8]). Still, there exists an important body of related work that already rely on such type of data sets [2, 3, 5–7, 14, 23, 26]. Individuals willing to know if their accounts appear in publicly leaked datasets may use online services such as `haveibeenpwned.com` or `sec.hpi.uni-potsdam.de/leak-checker`.

We would also like to emphasize our ethics regarding identifier providers. While we use publicly available data sets leaked from real organizations, our conclusion are not targeted against these organization. Our conclusions apply to *any* identifier provider using password-equivalents. Even though it is easy to reconstruct which data set we used, we anonymized the names of the related organizations or companies in this paper.

## 3 Description of the databases

In this Section, we describe the databases that we use for our study. We use four leaked password databases that we call $A$, $B$, $C$ and $D$. Table 1 summarizes some characteristics of these data sets. We set emphasis on the database $D$ as it is our main data set for this paper.

### 3.1 Data set $D$

In October 2013, a password and identifier database - denoted $D$ in the rest of the paper - was stolen from a software company and publicly released. At the time of its release, $D$ was the largest data set in its kind, with $153 \times 10^6$ identifiers (including email addresses) and $130 \times 10^6$ encrypted passwords. The

company quickly reacted by warning users and locking accounts. Anticipating contagion due to password reuse [9, 5], other identifier providers promptly asked their users to change their password.

$D$ was probably used by an authentication server used to access numerous products and services offered by the software company. $D$ covers a long time span of 12 years; the first identifiers were created in 2001. It seems that are very large and diverse set of services and applications of that company relied on the identifiers and passwords in $D$. While we do not know the exact list of services and applications that use $D$, they certainly include many standard applications provided by this software company. Users showing up in $D$ may also just have tried once an application, on a PC, on a phone, on a tablet, or registered to some web service (possible third party). Because of the above reasons a given user might have multiple identifiers and forgotten identifiers in $D$.

Analysts focused on password retrieval from $D$. Despite 3DES encryption, some passwords could be recovered because of three main reasons: (i) $D$ contains user provided hints in the clear, (ii) the passwords are encrypted with an unsalted 3DES, allowing comparison across different users, (iii) the encryption mode is Electronic Code Book, allowing the comparison of independent ciphertexts blocks of 8 characters. This combination of factors leads to an online ''crossword'' game for retrieving weak passwords[2]. $D$ has long been searchable through sites like `pastebin.com` and it is still accessible through peer-to-peer downloads.

The raw file contains 153 004 874 lines. We removed irregularities such as absurdly long or short lines, empty lines every 10 000 records, etc. In order to ease subsequent searches, we normalized the fields. Figure 2 shows the result of the normalization. The password equivalents in $D$ have the following structure: $pwdl = 3DES(left, S)$, $pwdr = 3DES(right, S)$ where $left$ is the first 8 characters of the clear password, $right$ is the next 8 characters. $S$ is a 3DES key only known by the software company. Only the owner of $S$ is able to formally verify clear passwords. In contrast, password equivalents made from unsalted hashes allow public verification. Without the key $S$, only an accumulation of evidences will reveal possible pairs of clear text passwords and password equivalents. Typical evidences are explicit *hint* such as: 'my password is frog35', 'frog + 7x5', '53gorf reverse'.

### 3.2 Other password databases

**Data set $C$ - a social network** The leaked data set contains 164x$10^6$ identifiers of a social network. The data set stores the users email address ($name@mail$) and a non-salted password hash. An entry in the data set $C$ is associated with a profile page on the social network.

**Data set $B$ - a social network** The leaked data set contains 1 057 596 identifiers of a social network. This data set stores the users email address

[2] See game `http://zed0.co.uk/crossword` and picture `http://xkcd.com/1286`

(*name@mail*) and a salted and hashed password. The data set includes URLs towards public profile pages (Facebook, Twitter, LinkedIn, Yahoo) if provided by the user.

**Data set *A* - an adult content site** The leaked data set contains 1 504 128 identifiers of an adult content site. This data set stores the users email address (*name@mail*) and a password in clear-text.

## 4 Privacy Attacks

In this section we describe three privacy attacks on $D$. We propose a set of classifiers that reveal potential links and secret links in Section 4.1 and Section 4.2 respectively. We also describe a method to deanonymize potentially secret identifiers in Section 4.3. Throughout this Section we depict our classifiers and methods using the examples of Table 2 ($k = 2$) and Table 3 ($k = 4$).

We evaluate, extend and discuss the presented classifier and methods in Section 5.

### 4.1 Revealing links

**Table 2.** Example case for 2-linkability.

| *uid* | *pwdl* | *pwdr* | *name* | *mail* | *hint* |
|---|---|---|---|---|---|
| 042...89 | gt...dfm | Qa...D | ingrid.computing | mycompany.com | as usual |
| 151...06 | gt...dfm | Qa...D | sexy_single_69 | somedatingsite.com | |

Let us consider the fictive case of Ingrid Computing as shown in Table 2. The privacy attacker will notice that only two identifiers in $D$ have the same password cipher "gt...dfm Qa...D". The attacker suspects a link between the two identities ingrid.computing@mycompany.com and sexy_single_69@somedatingsite.com. Both identifiers may of course relate to different persons, in which case the attacker makes a false positive in assessing a link. A motivated attacker may use external sources (search engines, OSN etc.) to collect more evidences, which is out of our scope. The above imaginary example depicts our first simple classifier for revealing links that we describe below.

**A classifier for 2-linkability :** The classifier tells that $L(x, y)$ (i.e. $x$ and $y$ are *linked*) if $\{x, y\} \in srp_2(D)$. $srp_2(D)$ is the set of identifiers having encrypted passwords appearing only twice in $D$.

The above classifier can be extended to k-linkability, i.e. to cases of password ciphers appearing exactly $k$ times in $D$. An illustrative example for $k = 4$ is

provided in Table 3.

**A classifier for k-linkability :** The classifier tells that $x_1, x_2 \dots x_k$ are *k-linked* if $\{x_1, x_2 \dots x_k\} \in srp_k(D)$. $srp_k(D)$ is the set of identifiers having encrypted passwords appearing exactly $k$ times in $D$.

### 4.2 Revealing secret links

Secret links are a subset of links. Coming back to the example shown in Table 2 the attacker might suspect a *secret link* since the *name* of both identifiers have nothing in common (have a small similarity). We propose the following classifier for secret links:

**A classifier for secret links for $k = 2$:** The classifier tells that $L(x, y)$ is a *secret link* if $\{x, y\} \in srp_2(D)$ and $jw(x, y) < s$ with a small $s$. $jw$ is the Jaro-Winkler similarity as defined in Section 2.2.

We also propose a classifier for secret links for cases where $k > 2$. We consider the cases where $k-1$ identifiers employ similar names and the remaining identifier is either a pseudonym of the same user or a different user. An example is provided in Table 3.

**A classifier for secret links for $3 \leq k \leq 9$:** We consider identifiers $x \in D$ such that $srp_k(x) \neq \emptyset$ and having the following properties: (i) $k-1$ identifiers in $srp_k(x)$ have similar *name*, for a chosen similarity and a threshold $s$, (ii) the remaining single identifier in $srp_k(x)$ does not have a similar name to any of the $k-1$ identifiers.

**Table 3.** Example data for a secret link with $k = 4$.

| *uid* | *pwdl* | *pwdr* | *name* | *mail* | *hint* |
|---|---|---|---|---|---|
| 05... | G...F | | ic.computing | email.xx | 1st cat |
| 05... | G...F | | 0699999996 | telco.xx | 1st cat |
| 06... | G...F | | computing.ic | telco.xx | kitty |
| 15... | G...F | | iccomputing | corp.xx | kitty |

We use the Stochastic Outlier Selection (SOS) [12] method to automate and build the above classifier. SOS is an unsupervised outlier-selection algorithm that provides an outlier probability for each data point. In our case the outlier is the remaining single identifier, which uses a *name* very different from the $k-1$ others. We apply SOS on $srp_k(x)$ and keep all sets of linked identifiers that exhibit a single and clear outlier. We conservatively consider an outlier to be an outlier if the SOS outlier probability is at least 0.98. Privacy attackers may adjust the threshold differently, according to their needs and resources.

### 4.3 Deanonymizing secret identifiers

*Secret links* can be used to deanonymize *secret identifiers*. Within the sets of identifiers that have a secret link, we search for sets of identifiers where at least one identifier reveals an identity, while the other linked identifiers do not. In the example of Table 2 the attacker might suspect that both identifiers relate to the same person, the first revealing a person's identity (the name of the person) while the second by itself does not reveal the person's identity (thus being a *secret identifier*). Similarly in Table 3, the phone number might be a *secret identifier* of a person which identity is revealed by the *name* of the other identifiers. We employ three heuristics, described below, to determine if an identifier reveals an identity of a person or not.

**Social network $B$:** The first heuristic uses the leaked data set $B$ of a social network. We consider that an identifier reveals an identity of a person if there exists an URL to a public profile page in the data set $B$. The data sets $D$ and $B$ both store the users email address ($name@mail$), allowing us to calculate joins of the two data sets.

**Social network $C$:** The second heuristic uses the leaked data set $C$ of a social network. An identifier in the data set $C$ is associated with a profile page on the associated social network, and we therefore consider that it reveals the identity of a person. The data sets $D$ and $C$ both store the users email address ($name@mail$), allowing us to calculate joins of the two data sets.

**US census:** The last heuristic verifies if the *name* part by itself reveals the identity of its owner. We use surnames provided by the US census[3]. We consider that an identifier reveals its owner's identity if the *name* contains a substring of at least four characters long equal to any surname occurring 100 or more times in the US. This heuristic is not very strict and may therefore include many false positives.

## 5 Evaluation

### 5.1 Evaluating classifiers for links

One objective of our analysis is to demonstrate $k$-linkability in $D$, and to provide an estimate of the probability $p$ that identifiers are actually linked. The main obstacle in such an analysis is the lack of ground truth. This prevents us from evaluating the results of our classifiers (e.g. calculate accuracies, false positives etc.) as it is done classically with machine learning problems. From a user perspective, the lack of such widely available ground truth in this domain is good news.

[3] See `https://www.census.gov/genealogy/www/data/2000surnames`

Instead of ground truth we use a set of heuristics on the password, the identifier name and the password hint. We also analyze the frequencies of these features to provide further evidence that two identifiers are in fact linked.

**2-linkability** We first evaluate the classifier for 2-linkability proposed in Section 4.1. The cumulated number of identifiers returned by this classifier is 13 507 724 (6 753 862 identifier pairs), representing 8.8% of identifiers out of $D$.

To estimate $p$ (the probability that two identifiers are actually linked) we use the heuristic that two identifiers link to the same person if the *name* fields are similar, i.e. $jw(x, y) \geq s$ or $ls(x, y) \geq s$. The strict equality ($s = 1$) provides a lower bound for $p$. The strict equality on the *name* field e.g. establishes that ingrid.computing@gmail.com and ingrid.computing@hotmail.com are the same person. The intuition is that the probability that two different users use the same *rare* password and the same *name* is almost zero. 10% identifier pairs have identical *name* in $srp_2(D)$. We consider this value as a pessimistic lower bound for $p$, i.e. $p \geq 0.1$.

By decreasing $s$ we obtain more optimistic values for $p$ (e.g. establishing that ingrid.computing@gmail.com and i.computing@hotmail.com are the same person). At the same time we may introduce more false positives. Figure 3 plots the cumulative distribution function of similarities of identifier pairs in $srp_2(D)$ for $ls$ and $jw$-similarities. Using a rather strict value for $s = 0.7$, we increase the proportion of linked identifiers to 23% with $ls$-similarity and 29% with $jw$-similarities. While we cannot provide more precise evidence, we strongly suspect that identifiers in $srp_2(D)$ are 2-linkable with probability $p$ greater than the pessimistic value 0.29.

We now compare the similarities of *name* between randomly sampled pairs out of $D$ (supposedly not linked) and identifier pairs in $srp_2(D)$ (supposedly linked). Figure 3 plots the cumulative distribution function of the similarities for both sets. We notice that the similarities are in general higher in $srp_2(D)$; the mean $ls$-similarity in $srp_2(D)$ is 0.42 versus 0.19 for random pairs. Similarly, the mean $jw$-similarity in $srp_2(D)$ is 0.58 versus 0.40 for random pairs. Finally, the proportion of random identifier pairs having identical *name* is in the range of 0.003%, compared to 10% in $srp_2(D)$. These numbers confirm the *name* is in general closer between identifier pairs in $srp_2(D)$, than any other random identifier pair.

As a further indirect evidence, we show that the propensity of a user to reuse passwords is much higher within $srp_2(D)$. We use the *hint* field to estimate the propensity of a user to reuse passwords. More precisely, we count the number of *hint* fields containing terms indicating password reuse: 'as usual', 'always', etc. See Annex 7.3 for the full list. The result is shown in Figure 4. Among the 66 493 790 identifiers with unique passwords within $D$, 435 842 identifiers (0.7%) have a 'as usual' kind of hint. Among the 13 507 724 identifiers that share their password exactly once with some other identifier, 173 272 (1.3%) have a 'as usual' kind of hint. The proportion almost doubles, confirming the higher propensity of

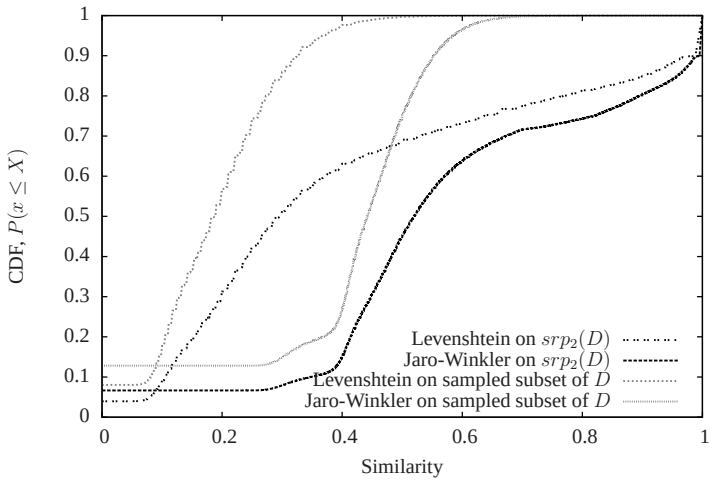



**Fig. 3.** Cumulative distribution function of similarities in $srp_2$ and in randomly sampled pairs of identifiers of $D$.

users in $srp_2(D)$ to reuse passwords.

In light of the above discussion, we propose a more accurate classifier for 2-linkability, i.e. the classifier has higher values for $p$, at the price of returning a smaller number of identifiers. The classifier tells that $L(x, y)$ if: $(x, y) \in srp_2(D)$ and $jw(x, y) \geq s$ for a similarity parameter $s$. With $s = 1$, we link 683 722 identifier pairs of $D$ with a $p$ close to 1. As discussed before, decreasing $s$ increases the number of linked identifiers but decreases $p$. The precision of this classifier can be further extended by adding the condition that the *hint* indicates password reuse.

**k-linkability** Figure 5 shows the number of *k-links* (being $|srp_k(D)|$) revealed by the classifier for k-linkability of Section 4.1. Table 4 provides the cumulative number of links for $k = 2 \ldots 9$. We can observe, that the number of revealed *k-links* gradually decreases with $k$. As discussed in Section 5.1, the probability $p$ that the corresponding identifiers are linked to a same user should decrease when $k$ increases. To demonstrate this trend we consider the propensity of a user to reuse a password (Figure 4). The ratio of *hint*s indicating password reuse is similar with $k = 3$ and $k = 2$. For $k > 3$ this ratio regularly decreases, indicating that $p$ also decreases.

The absolute numbers of *k-links* of Figure 5 are difficult to interpret, particularly because it is difficult to estimate the probability $p$. Still, the results of the *k-link* classifier can be further filtered and refined to reveal *secret links* and *secret identifiers*.

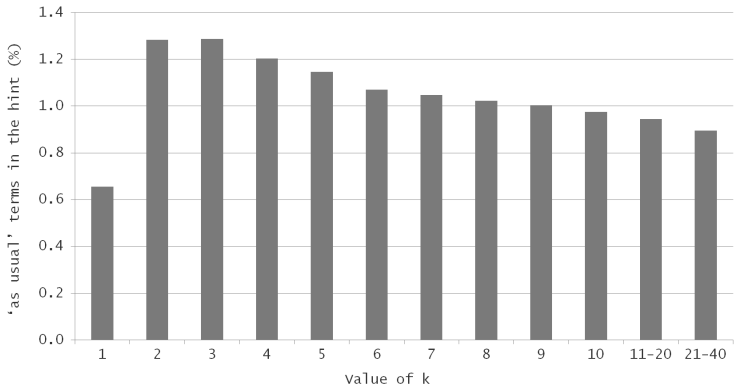



**Fig. 4.** Percentage of ''as usual'' terms in the *hint*, as a function of $k$ in $k$-linkability. $k$ takes values in $\{1, 2, \ldots, 10, 11-20, 21-40\}$.

**Table 4.** Cumulative number of *links*, *secret links* and *secret identifiers* in $D$ for $k$ between 2 and 9.

| | |
|---|---|
| *#links* | 11 038 079 |
| *#secret links* | 1 937 634 |
| *#secret identifiers using US census* | 763 348 |
| *#secret identifiers using social network C* | 348 892 |
| *#secret identifiers using social network B* | 4 003 |
| in comparison: size of $D$ | 153 004 874 |

### 5.2 Evaluating classifiers for secret links and secret identifiers

We now evaluate the classifier for *secret links* and *secret identifiers* proposed in Section 4.2 and Section 4.3. *Secret links* and *secret identifiers* are supposed to be *secret* and it is even more difficult to find ground truth than with *links* (e.g. the secret links will in general not appear on Google, Facebook or LinkedIn profile pages). We therefore first provide global results and numbers and then focus on a corner-case experiment consisting in deanonymizing role based emails. We also intersect the revealed *secret identifiers* with external data sets ($A$) and discuss the potential impacts.

**Secret links and secret identifiers global results** For *secret links*, we set $s = 0.4$ (as defined in Section 4.2) and therefore require that $jw(x, y) < 0.4$. This threshold corresponds to the first ''elbow'' in Figure 3. Doing so, we estimate that an attacker would reveal 1 million potential *secret links*. Figure 5 also shows the number of revealed potential *secret links* with $k > 2$; Table 4 provides the cumulative numbers. While it is difficult to assess the $p$ of this classifier, we know that we can increase $p$ by adding the condition that the *hint* indicates password

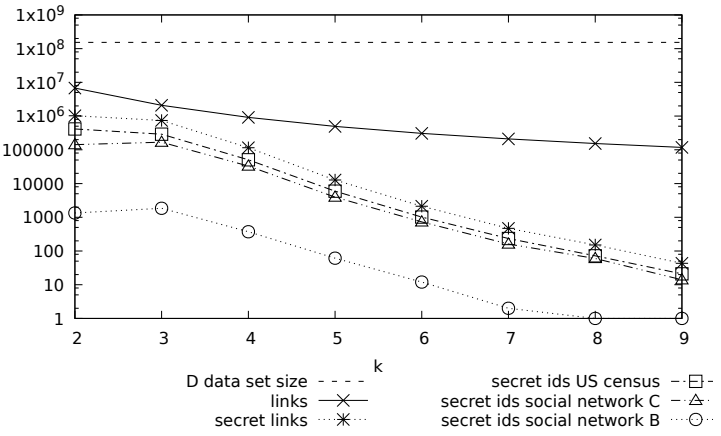



**Fig. 5.** Number of *links*, *secret links* and *secret identifiers* in $D$ for $k$ between 2 and 9.

reuse. Figure 5 and Table 4 further shows the number of *secret identifiers* we could deanonymize in $D$, according to the three deanonymization heuristics proposed in Section 4.3.

**Deanonymizing role-based emails** The classifiers may discover real names behind generic email addresses like support, admin, security, etc. An attacker can use this knowledge to bypass an 'administrator' or 'support' email address and directly contact the real person in charge. For this application, we select pairs of identifiers in $srp_2(D)$ such that: (i) one *name* is generic (see Annex 7.3), (ii) both identifiers have the same *mail* part, (iii) the *mail* part is rare within $D$ (less than 100 occurrences in our experiment). From such pairs, the privacy attacker can automatically generate a human readable statement such as: ''Ingrid Computing is 'sysadmin' at this-company.com''. The fact that linked identifiers have the same rare *mail* part reinforces the link, at least from a Bayesian perspective. The above method generates 25 253 statements involving at least one generic identifier. Among those, 2 858 statements involve a *name* part with a separator (".","_") and forenames and lastnames that are not reduced to a single letter.

**Intersecting with other databases** We demonstrate the impact of our attack by deanonymizing *secret identifiers* in the data set $A$, a data set related to adult content. We use the $A$ since we expect to find more users that would like to remain anonymous with a service related to adult content. This is confirmed by the numbers of identifiers revealing person identities using the *US census*, *social network C* and *social network B* heuristics (see Table 5). The proportion of identifiers that reveals person identities is systematically smaller in the data set $A$.

**Table 5.** Proportion of identifiers in $D$ and $A$ revealing person identities according to different heuristics.

| | *US census* | *social network C* | *social network B* |
|---|---|---|---|
| $D$ | 93.83 % | 8.33 % | 0.07 % |
| $A$ | 78.03 % | 3.91 % | 0.02 % |

**Table 6.** Number of deanonymized secret identifiers in $A$, and number of secret links according to different criterions

| *Deanonymized secret identifiers* | | | *Secret links* | | | |
|---|---|---|---|---|---|---|
| *US census* | *social network C* | *social network B* | *all* | *corporate* | *gov* | *univ* |
| 851 | 337 | 5 | 2979 | 3 | 4 | 104 |

We deanonymize a *secret identifier* in the data set $A$ by (i) extracting all *secret identifiers* in $D$ and (ii) keeping only the *secret identifiers* (*name@mail*) that also appear in $A$. The data sets $A$, $B$, $C$ and $D$ all include email addresses (*name@mail*), allowing us to calculate joins. Table 6 reports the number of deanonymized identifiers.

We further highlight the existence of embarrassing *secret links*. In Table 6, we report the number of *secret links* between an identifier in $A$ and identifiers that verify a set of criteria: *(all)* no restriction on the mail address, *(corporate)* corporate mail addresses from major companies, *(gov)* mail addresses from government agencies, *(univ)* mail addresses from universities.

## 6 Related Work

Related work focuses on password cracking, password strength, password user studies and deanonymization of public data sets.

The most common password cracking attacks are the brute-force and dictionary attacks using popular tools such as John the Ripper. Many improvements for password cracking have been proposed: using rainbow tables [22], using Markov models [18], using probabilistic context-free grammars [26], etc.

Some works try to assess or measure the strength of a password [6, 15, 24, 1, 2, 13, 7]. In this context, password meters are supposed to help users to improve their password. However, Ur et al. [25] show that in general password meters only marginally increase the resistance to password cracking. Only very strict password meters tend to increase the password strength [25, 3]. Password popularity is also used to measure the strength of a password [24, 7, 14]. To strengthen a password, Schechter et al. [24] use the quite simple idea of discouraging the use of popular passwords. This latter approach is clearly beneficial for the privacy attacker of this work. The above works often use well-known password data sets to evaluate their performance.

Other work considered user behavior regarding passwords [15, 9, 5]. [9, 5] study the problem of password reuse across sites. Both show that the reuse

of the same or a similar password is a predominant practice for end-users. In particular, [5] studies how users transform their password for different online accounts. Both papers focus on an attacker breaking passwords, e.g. [5] builds a password guessing algorithm based on the observed user behavior. These works do not consider the privacy attacker which does not require to break passwords.

Most privacy attacks focus on the deanonymization of social networks and rating systems. [19] deanonymizes the public Netflix data set, by matching movie ratings provided by users of the Internet Movie Database. [20] re-identifies users of an anonymized Twitter graph, matching them against Twitter and Flicker identifiers. [17] identifies anonymous online authors by comparing writing styles. [16] links reviewers of community review sites. We consider a radically different type of data set that has not been studied in terms of privacy so far.

To the best of our knowledge, the work that comes closest to ours is [23]. The authors use identifier names to link or uniquely identify users. They further leverage textual similarities between identifier names for estimating the linkability probabilities. Our work is different as (i) we use encrypted information rather than textual information and (ii) we link to secret identifiers that are -- by definition -- very dissimilar from their linked identifiers.

## 7 Discussion and Conclusion

We presented linkability attacks based on password equivalents in leaked identifier and password data sets. The attacks do not require breaking a single password, and the efficiency increases with the password strength. Having no ground truth, which is expected in this domain, we provided indirect assessment of the performance of our classifiers. We demonstrated the consequences of our attack by showing that a privacy attacker can reveal sensitive private information such as secret identifiers. In particular, we evaluated how privacy attackers can deanonymize secret identifiers of users of adult content sites. State of the art attacks analyzing online social networks do not reveal this kind of information.

### 7.1 Tractability of privacy attacks

We would like to emphasize several risks for people's privacy. First, the presented privacy attacks require little computation resources. For instance, the $k$-linkability analysis on $D$ took only 400 cumulated computation hours. The complexity of most treatments does not exceed $O(n.log(n))$. The attacker does not need to break any password, which saves a lot of resources. Further, the attacks can be performed using publicly available data sets. There is no need to crawl social networks or to have access to a social network graph. These two facts make our attacks tractable to most individuals without requiring any specific privileges or computing power. Finally, $D$ is much larger than other data sets in this domain. This allows retrieving a fair amount of results, typically thousands, even when using multiple refinement requests. The number and size of publicly available data sets of that kind tends to increase, meaning that the number of retrieved results will also further increase over time.

### 7.2 Mitigations

The mitigations and countermeasures are rather classical. End-users achieve best results in terms of both privacy and security by using a strong and different password for each service. Since it might be difficult for a user to remember all these passwords, we recommend users to segment linkability according to their estimated privacy needs. Users should use unique passwords for the few services that they never want to be linked to. For other non-privacy critical services, users may use a password based on one single root (e.g. *frog35!*), and prefix the password with a character or string related to the service (e.g. *FB* for Facebook, *LI* for LinkedIn). This ''poor man's salt'' does not reinforce the security of the password, but decreases the impact of linking attacks. Password managers that generate randomized passwords also provide an efficient countermeasure. Finally, identifier providers should use salted hashing functions. These recommendations have been published several years ago and still, numerous leaked files reveal bad practices. In addition, we encourage identifier providers to encrypt both the hints and the email addresses. Obviously the hints are private, while massively leaked email addresses are a gift to spammers. Finally, identifier providers should avoid incremental uid's and use random numbers [10].[4]

**Table 7.** Probable history of a user w.r.t data set $D$.

| *uid* | *pwdl* | *pwdr* | *name* | *mail* | *hint* |
|---|---|---|---|---|---|
| 06...83 | hc...si | | joe.target | corp1.com | |
| 10...68 | sj...f2 | Tr...G | joe_target | corp2.com | |
| 16...80 | sj...f2 | Tr...G | tryjoe | isp.com | usual |
| 17...22 | Fg...st | | tryjtarget | corp3.uk | other |

### 7.3 Future work

We found several cases where additional private information can be inferred from the available data sets. For instance, a privacy attacker could deduce people ''histories'' from the set of successive identifiers of a same person. Table 7 shows one example. Using time reconciliation this history reads: ''In 2001, Joe was at corp1, he joined corp2 before mid-2008, then he went to corp3 before 2012''. Building such histories requires linking identifiers through names [23] , in addition to the links established through passwords. The first entry in Table 7 is linked to the second via distances introduced in [23]. The second entry is linked to the third entry via the password. The fourth item is linked to all others via a combination of both techniques.

[4] The *uid* of $D$ increases monotonically with the time of creation of the identifier. It allows the reconstruction of a timeline, by e.g. using creation dates of some identifiers or by searching in the fields *name* and *hint* for events having a worldwide notoriety.

## Acknowledgements

We thank the Program Committee and reviewers for the many valuable comments that significantly improved the final version of this paper.

## References

1. J. Bonneau. The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *IEEE Symposium on Security and Privacy*, 2012.
2. J. Bonneau. Statistical metrics for individual password strength. In *20th International Workshop on Security Protocols*, April 2012.
3. C. Castelluccia, M. Dürmuth, and D. Perito. Adaptive password-strength meters from markov models. In *Network and Distributed System Security (NDSS) Symposium*, 2012.
4. W. W. Cohen, P. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *KDD Workshop on Data Cleaning and Object Consolidation*, 2003.
5. A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang. The Tangled Web of Password Reuse. In *Network and Distributed System Security (NDSS) Symposium*, 2014.
6. M. Dell'Amico, P. Michiardi, and Y. Roudier. Password strength: An empirical analysis. In *IEEE INFOCOM*, 2010.
7. W. Ding and P. Wang. On the implications of zipf's law in passwords. In *ESORICS*, 2016.
8. S. Egelman, J. Bonneau, S. Chiasson, D. Dittrich, and S. Schechter. Its not stealing if you need it: A panel on the ethics of performing research using public data of illicit origin. In *3rd Workshop on Ethics in Computer Security Research*. Springer, 2012.
9. D. Florencio and C. Herley. A large-scale study of web password habits. In *ACM WWW*, 2007.
10. S. Gambs, O. Heen, and C. Potin. A Comparative Privacy Analysis of Geosocial Networks. In *4th ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS*, SPRINGL '11, 2011.
11. S. Halevi and H. Krawczyk. Strengthening digital signatures via randomized hashing. In *Advances in Cryptology - CRYPTO*, 2006.
12. J. Janssens, F. Huszßr, E. Postma, and J. van den Herik. TiCC TR 2012-001, Stochastic Outlier Selection. Technical report, Tilburg University, 2012.
13. P. G. Kelley, S. Komanduri, M. L. Mazurek, R. Shay, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, and J. Lopez. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In *IEEE Symposium on Security and Privacy*, 2012.
14. D. Malone and K. Maher. Investigating the distribution of password choices. In *ACM WWW*, pages 301--310. ACM, 2012.
15. M. L. Mazurek, S. Komanduri, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, P. G. Kelley, R. Shay, and B. Ur. Measuring password guessability for an entire university. In *ACM CCS*, 2013.
16. M. A. Mishari and G. Tsudik. Exploring Linkability of User Reviews. In *ESORICS*, 2012.

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 *IEEE Symposium on Security and Privacy*, 2012.
18. A. Narayanan and V. Shmatikov. Fast dictionary attacks on passwords using time-space tradeoff. In *ACM CCS*, 2005.
19. A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *IEEE Symposium on Security and Privacy*, 2008.
20. A. Narayanan and V. Shmatikov. De-anonymizing social networks. In *IEEE Symposium on Security and Privacy*, 2009.
21. M. E. Newman. Power laws, pareto distributions and zipf's law. *Contemporary physics*, 46(5):323--351, 2005.
22. P. Oechslin. Making a faster cryptanalytic time-memory trade-off. In *Advances in Cryptology - CRYPTO*. 2003.
23. D. Perito, C. Castelluccia, M. A. Kaafar, and P. Manils. How unique and traceable are usernames? In *PETS*, 2011.
24. S. Schechter, C. Herley, and M. Mitzenmacher. Popularity is everything: A new approach to protecting passwords from statistical-guessing attacks. In *USENIX HotSec*, 2010.
25. B. Ur, P. G. Kelley, S. Komanduri, J. Lee, M. Maass, M. Mazurek, T. Passaro, R. Shay, T. Vidas, L. Bauer, et al. How does your password measure up? the effect of strength meters on password creation. In *USENIX Security*, 2012.
26. M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek. Password cracking using probabilistic context-free grammars. In *IEEE Symposium on Security and Privacy*, 2009.

## Appendix

### Terms for 'as usual'

always, usual, the rest, for all, normal, same as, standard, regular, costumbres, siempre, sempre, wie immer, toujours, habit, d'hab, comme dab, altijd.

### List of generic email addresses

abuse admin administrator contact design email info intern it legal kontakt mail marketing no-reply office post press print printer sales security service spam support sysadmin test web webmaster webmestre.