Analysis of Publicly Leaked Credentials and the Long Story of Password (Re-)use

David Jaeger, Chris Pelchen, Hendrik Graupner,
Feng Cheng, and Christoph Meinel

Hasso Plattner Institute, University of Potsdam, Germany
{firstname}.{lastname}@hpi.de

Abstract. Nowadays, identity breaches are happening almost on a daily basis. Just recently, hundreds of millions of identities were leaked from services like LinkedIn, MySpace and VKontakte. Undoubtedly, these breaches constitute a major threat because victims might fall to identity theft. As part of our warning service for victims of these breaches, we have gathered and normalized most of the publicly available breaches and could assess nearly one billion credentials. Apart from our security awareness service, the large amount of real world credentials allows to create comprehensive and realistic password statistics. In this paper, we introduce multiple comprehensive statistics on the use of passwords based on the gathered data. We especially focus on the often mentioned, but rarely researched, issue of password reuse and reveal the regional differences in password selection. We are confident that the analysis of such a large amount of real-life credentials is novel to existing studies on passwords, which were limited to thousands or a few million credentials, at most. For the first time, a realistic view on password reuse can be given.

Keywords: identity leak, data breach, password reuse, security awareness

1 Introduction

The last few months have shown that data breaches are more prevalent than ever and there is no real protection against them. Almost daily, reports about major breaches can be found in the news. In the first half of 2016 alone, multiple breaches of popular services were revealed, which put hundreds of millions of innocent service users at the risk of identity theft. Among these affected services are big players like VKontakte (≈ 93m users) [1], LinkedIn (≈ 164m users) [2], MySpace (≈ 360m users) [3], Tumblr (≈ 65m users) [4] and Twitter (≈ 33m users) [5]. Although the breached services responded quickly by resetting the passwords of their compromised accounts, many users are still at risk as they reuse the same password across various services. The relevance of this negligence could be observed directly after the previously mentioned data leaks, when news about compromised social-media accounts from quite a few well-known celebrities came up [6, 7]. All of these accounts were taken over by making use of credentials from the LinkedIn or MySpace-leak.

Of course, password reuse is not only a problem of celebrities, but is a common phenomenon for password selection of many users. Even worse, as credentials are still the main method of authentication on computer systems, password reuse has become an important attack vector for miscreants to take over accounts. According to a report by Verizon[8], as much as 1,095 out of 1,462 reported security incidents involved stolen passwords, also covering password reuse, as attack vector.

Although the problem of password reuse is widely known and awareness for it is raised, there is no comprehensive empirical analysis on how many users are really reusing their passwords across multiple services. A more specific type of password reuse, the slight variation of a base password for each service, is even less analyzed but also relevant. So how big is this problem in the real world? Instinctively, it seems that password reuse is very common among users. According to a survey [9] on password habits with 1,200 participants by the company CSID, around 61% of users reuse their passwords.

In this paper, we shed light on the phenomenon of password reuse and the general concepts of password use by analyzing a large dataset of real-world credentials from publicly accessible identity breaches. This enables us to create unbiased statistics on passwords and reflects how users choose their passwords for productive services.

The organization of this paper is as follows. In Section 2 related work on the analysis of leak data from academic research is mentioned. At this point the focus is on the analysis of regional password and the analysis of password reuse. The following Section 3 presents our Identity Leak Checker Service and its workflow. Section 4 covers the actual analysis of passwords. First, we name and describe the data breaches we selected for our analysis. After that we explain our performed analysis techniques and present the respective results. We focus on a general analysis on passwords, the analysis on password reuse and the analysis on country-specific passwords. The following Section 5 evaluates the possibilities of improving the efficiency of password cracking by password reuse. After that, we conclude in Section 6 and provide an outlook of our next steps as well as future work to be done.

2 Related Work

The analysis of passwords has a long history and has been covered in a large amount of scientific works. In our paper, we are going into the direction of password analysis from massive identity leaks. Looking at existing work, we believe that this is a rather new approach and not well covered. Nevertheless, we have found some related work on leaked data and our concrete password analysis approaches.

2.1 Analysis of Leak Data

The existing research on identity leaks mainly covers the analysis of passwords that can be extracted from them. We have analyzed the origin of identity leaks [10] and the password routines used for the storage of passwords in the leaked services [10, 11]. Both of our works also illustrate the normalization of the provided information in leaks. The security researcher Troy Hunt is researching the ecosystem of identity leaks and regularly publishes his findings on passwords and leaks in his blog[1]. There is also a variety of works that focus on the analysis of leaked credentials from so called paste-pages, such as *pastebin.com* [12, 13, 14]. They show that a considerable large amount of credentials can be obtained from public web-sites and it is easy to find sensitive users from big companies and government agencies among them.

2.2 Analysis of Regional Passwords

The analysis of passwords from different regions is not well covered in research, probably because password studies are not broad enough to have a representable sample and previously analyzed password lists do not have information on the origin of a password.

Jaeger et al. have conducted [10] a password analysis on leaks and have extracted top passwords from pure Chinese leaks. They found out there are indeed differences to international passwords. However, their analysis was only limited to Chinese passwords. It would be interesting to also have a look at other countries, especially those with non-latin character sets, such as countries with Cyrillic alphabets and Arabic languages. Dell' Amico et al. have looked [15] into the cracking of passwords with dictionaries from different countries and showed that language-specific dictionaries have higher success rates when cracking passwords of these languages.

[1] Blog of Troy Hunt - `troyhunt.com`

For concrete regional word lists, the collections of Packet Storm Security [2] and OpenWall[3] are very interesting.

2.3 Analysis of Password Reuse

The reuse of passwords across multiple services and accounts has only been researched on a small scale, yet. On the one hand, there are user surveys about the reuse of passwords, such as the one from CSID [9]. Nevertheless, the general problem of these surveys is the relatively small sample size and a bias in the responses, as people might not want to reveal their real password habits to the public. On the other hand, there are statistics on real-world datasets. The ones we could find were from Hunt [16] and Das et al. [12]. Hunt's analysis merely covers two leaks with 88 users in common and shows a password reuse of 33%. Das' analysis covers 10 leaks with 6077 users in common and shows a password reuse of 43%. Nevertheless, the half of these leaks are not verified and might originate from credential stuffing[4] with 100% reuse.

3 Identity Leak Checker Service

As the problem of identity leaks is getting more serious and victims often do not even know when they are affected by identity theft, we have created a free-of-charge web service[5] where Internet users can check whether their account data appears in public data leaks. We have created this service as an awareness service that warns the victims of identity theft but also gives hints on the proper use of passwords on the Internet. Since the start of our service in May 2014, more than 2.5 million people have checked whether their data has been found in public identity leaks. From these, we could warn more than 200.000 of leaked data and provided countermeasures. At the moment, we have analyzed around 100 leaks with over 1 billion user records. To provide the leaked data for querying in our service, we have established a privacy-aware processing workflow for leaks that is described in the following.

[2] Packet Storm Wordlists - `https://packetstormsecurity.com/Crackers/wordlists/`

[3] OpenWall Wordlists - `http://download.openwall.net/pub/wordlists/languages/`

[4] password guessing with already leaked credentials on different services

[5] Identity Leak Checker Service - `https://sec.hpi.de/ilc`

3.1 Workflow

An overview of our workflow is shown in Figure 1. At first, identity leaks are gathered from various different sources, such as dedicated forums, paste pages, file sharing pages, and social network announcements. A detailed overview of these sources can be found in the work by Jaeger, Graupner, et al. [10]. On these raw dumps of identity data, a normalization routine is applied that first detects the kind of format used in the leak [11], i.e. CSV, SQL, or others, and then tries to convert this format into a common CSV-format. Although the data itself is normalized, it is still not known what kind of data is in the leak. Therefore, the normalization is followed by an interpretation step that categorizes the raw data into identity-specific data. In the last step before the persistence into the database, the identity-specific data is anonymized. This anonymization covers the hashing of the email address and its top-level domain as well as the replacement of concrete personal data, such as address, name, or phone numbers, with a category tag.

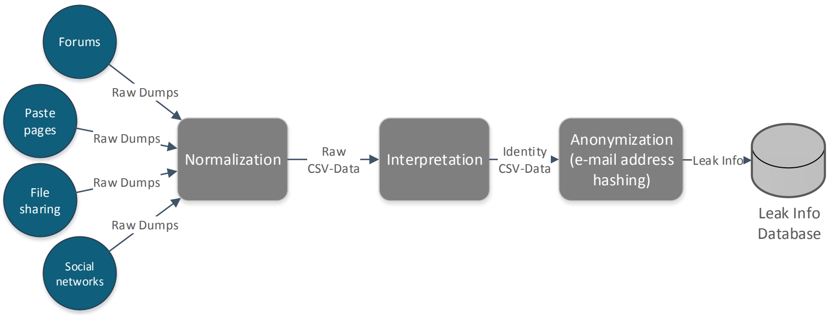


Fig. 1. Processing Workflow of our Identity Leak Checker Service

Using this anonymization routine, it is possible to perform analysis on passwords in relation to the owner's email address without directly having access to the email address and its identity behind it. Additionally, other personal information is not kept and cannot be extracted after this process. Surely, it would still be possible to recover the records of a known email address, but it has to be considered that these records would be accessible anyhow from the public leaks. We just want to prevent direct and easy access to all available leak data and keep the stored information as limited as possible.

4 Analysis on Passwords

Our research and analysis of identity leaks for the Identity Leak Checker Service does not only allow us to warn victims, but also to conduct large-scale password analysis on anonymized data. In this section, we first introduce the data sources we have used and then derive statistics on the large amount of included credentials.

4.1 Leak Overview

For our analysis of credentials, we have gathered and normalized the biggest publicly available identity databases to date. Altogether, we have found 31 identity leaks with most of them having nearly 1 billion credentials in them. Table 1 summarizes the most important information about the analyzed leaks.

ID	Name	Passw. routine	Accounts with passw.	Leak date
1	000webhost.com	`$p`	15 035 687	≈ Mar. 2015
2	17.media	`md5($p)`	3 824 575	≈ Sep. 2015
3	51cto.com	`md5(md5($p).$s), md5($p)`	3 923 449	≈ Dec. 2013
4	7k7k.com	`$p`	9 231 185	≈ Oct. 2011
5	aipai.com	`md5($p)`	4 529 928	≈ Apr. 2011
6	ashleymadison.com	`bcrypt($p)`	36 140 796	≈ July 2015
7	badoo.com	`md5($p)`	122 730 419	≈ June 2016
8	csdn.net	`$p`	6 425 905	≈ Oct. 2011
9	duduniu.cn	`$p`	14 192 866	≈ Aug. 2011
10	gawker.com	`des($p)`	487 292	≈ Dec. 2010
11	gmail.com	`$p`	4 925 994	≈ Sep. 2014
12	imesh.com	`md5(md5($p).$s)`	51 308 651	≈ Sep. 2013
13	ispeak.cn	`$p`	8 294 278	≈ Apr. 2011
14	linkedin.com	`sha1($p)`	112 275 414	≈ Feb. 2012
15	mail.ru	`$p`	5 269 103	≈ Sep. 2014
16	mate1.com	`$p`	27 402 581	≈ Feb. 2016
17	mpgh.net	`md5(md5($p).$s)`	3 119 180	≈ Oct. 2015
18	myspace.com	`sha1($p)`	358 986 419	≈ 2008
19	naughtyamerica.com	`md5($p)`	989 401	≈ Apr. 2016
20	nexusmods.com	`md5(md5($s).md5($p))`	5 918 540	≈ Dec. 2015
21	r2games.com	`md5(md5($p).$s), md5($p)`	11 758 232	≈ Oct. 2015
22	renren.com	`$p`	4 392 208	≈ Nov. 2011
23	sprashivai.ru	`$p`	3 472 645	≈ May 2015
24	taobao.com	`$p`	14 769 995	≈ Jul. 2015
25	tianya.cn	`$p`	29 642 564	≈ Nov. 2011
26	twitter.com	`$p`	26 121 984	≈ June 2016
27	vk.com	`$p`	92 144 526	≈ 2012
28	weibo.com	`$p`	4 529 994	≈ Dec. 2011
29	xiaomi.com	`md5(md5($p).$s)`	8 281 358	≈ May 2014
30	xsplit.com	`sha1($p)`	2 990 112	≈ Nov. 2013
31	yandex.ru	`$p`	1 186 565	≈ Sep. 2014
Total accounts with email addr.: 994 301 846 , **Total distinct email addr.:** 884 460 979				

Table 1: Analyzed identity leaks ($p - clear password, $s - salt)

All above 31 identity leaks contain 994 301 846 different credentials. We define a credential as a data record with an email address and a cleartext password or hash. We have removed all records where no valid email address or a password in cleartext or hashed form is given. All the credentials originate from as much as 884 460 979 different email addresses. The white color for the rows indicates that the data seemingly originates from an extracted database of a service. Gray rows indicate unverified service leaks, i.e. they could also be a collection of credentials, such as from password guessing or phishing. We will show later how this distinction is important for the analysis of password reuse.

The second column lists the used password routine within the leak. A summary of these routines can be found in Table 2.

Hash routine	Common name	# of leaks	# of dumps
`$p`	cleartext	16 (≈ 51.6%)	6 (≈ 28.5%)
`md5($p)`	MD5	4 (12.9%)	4 (≈ 19.0%)
`sha1($p)`	SHA-1	3 (9.7%)	3 (≈ 14.3%)
`des($p)`	descrypt	1 (≈ 3.2%)	1 (≈ 4.8%)
`md5(md5($p).$s)`	vBulletin-Hash	5 (≈ 16.1%)	5 (≈ 23.8%)
`md5(md5($s).md5($p))`	MyBB-Hash	1 (≈3.2%)	1 (≈ 4.8%)
`bcrypt($p)`	bcrypt	1 (≈ 3.2%)	1 (≈ 4.8%)

Table 2. Password routines of all identity leaks

Surprisingly, there are still ≈ 59% of the services that store their passwords in cleartext or insecure MD5 or SHA1. Another ≈ 32% use simple hashing with salts, but not make use of rounds or similar security enhancements. Only one source, i.e. *Ashley Madison*, made use of strong hashes, although a report [17] revealed that this site also used weak MD5 hashes for around half of their accounts.

For the leaks that do not originate from a service database, it cannot be ensured that the data is legitimate. After analyzing some of the emails of these credential collections, we found out that some of them might be related, meaning that some very rare passwords appear across multiple of these leaks. The main reason for this could be that the credentials of one of these leaks could have been guessed from the credentials of a previous leak. To handle this relation, we introduce the concept of a *leak group*. By default, each leak has its own leak group. However, related leaks share the same leak group. This means that all credentials in one leak group are independent of the credentials of other leak groups. *Chinese1* is a group of leaks that are related to the Tianya-leak in late 2011 [18]. We believe that the two other leaks from 7k7k and ispeak.cn are composed from the credentials of the Tianya-leak, because they contain a high amount of unique Tianya-related passwords. The *Chinese2* group is very suspicious, because as much as 98% of all credentials in *weibo.com* and *renren.com* are the same. Considering the timely proximity of all leaks in both leak groups and their Chinese origin, it is quite possible that even both leak groups are related.

4.2 Password Analysis Procedure

In the following subsections, we focus on the analysis of the leaks' passwords. As Table 1 revealed, many of the leaks do not have the passwords in cleartext, but use password hashes. To be able to analyze these passwords, the cleartext passwords behind these hashes have to be found. There are multiple ways to lookup the cleartext of a hash, as listed below.

1. Lookup the hash in a *rainbow table*
2. Lookup the hash in *Google*, because many people have tried to crack simple hashes already
3. Lookup the hash in one of several *hash cracking web-pages*, such as HashKiller[6]
4. Download lists of *pre-cracked passwords* for a leak, such as from hash-cracking forums

All of the methods do not guarantee that the cleartext of a hash can be found. Still, to get the most passwords revealed, we first checked hashes against lists of pre-cracked passwords for the corresponding leak. After that, we created a word list from all cleartext passwords of all our leaks and have used hashcat[7] with the `dive`-ruleset to find some more passwords of the remaining hashes.

After looking up as many passwords as possible and combining them with the given cleartext passwords, we ended up with the numbers in Table 3. All in all, we could recover $\approx$ 848 million credentials with cleartext passwords having $\approx$ 320 million different passwords. Comparing these numbers to the total amount of credentials, it means that there are cleartexts for around 85% of all credentials.

[6] HashKiller - `hashkiller.co.uk`

[7] Hashcat - `http://hashcat.net`

Table 3. Credentials with cleartext passwords and percentage of recovered encrypted password, - was used for cleartext only leaks

Name	Clear cred.	Rec.	Name	Clear cred.	Rec.
000webhost.com	15 035 687	-	mpgh.net	247 499	8%
17.media	2 709 893	71%	myspace.com	328 152 578	91%
51cto.com	2 228 479	67%	naughtyamerica.com	911 781	92%
7k7k.com	9 231 185	-	nexusmods.com	2 691 088	45%
aipai.com	2 221 875	49%	r2games.com	364 927	3%
ashleymadison.com	2 559 028	8%	renren.com	4 392 208	-
badoo.com	114 090 491	97%	sprashivai.ru	3 472 645	-
csdn.net	6 425 905	-	taobao.com	14 769 995	-
duduniu.cn	14 192 866	-	tianya.cn	29 642 564	-
gawker.com	439 449	90%	twitter.com	26 121 984	-
gmail.com	4 925 994	-	vk.com	92 144 526	-
imesh.com	15 908 834	32%	weibo.com	4 529 994	-
ispeak.cn	8 294 278	-	xiaomi.com	1 167 052	14%
linkedin.com	104 955 280	93%	xsplit.com	2 904 588	97%
mail.ru	5 269 103	-	yandex.ru	1 186 565	-
mate1.com	27 402 581	-			
Total cleartext cred.: 848 590 922 , **Cleartext passwords:** 320 201 615					

4.3 General Analysis

To give a general overview on all passwords, we create some common statistics.

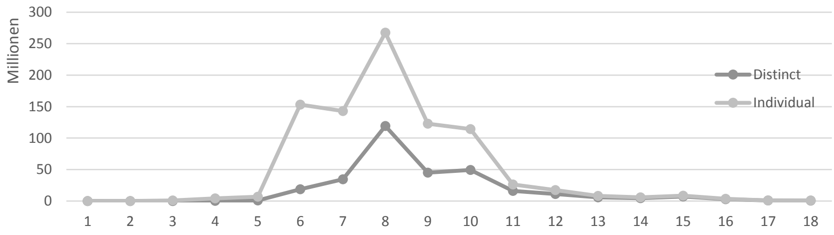
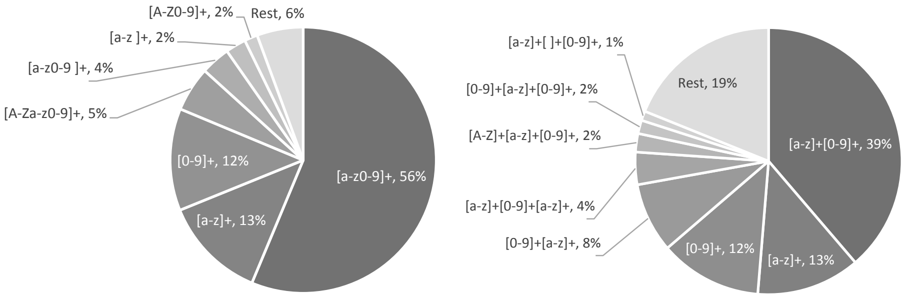


Fig. 2. Distribution of password lengths (distinct - each password only once, individual - password used by a user in a leaked source)

Length Distribution The length distribution of passwords is depicted in Figure 2. It reveals that most passwords have 8 characters, for distinct as well as individual passwords. This could be explained with the common recommendation to use passwords with at least 8 characters. Also, a very large part of all passwords is in the range between 6 and 12 characters. This can help when performing length-based password cracking attacks, such as the PRINCE-attack [19] or general brute-forcing. It has not be noted, that the diagram can be slightly biased for longer passwords, as some hashed passwords with these lengths might not have been recovered with cracking.



(a) Character classes

(b) Character sequences

Fig. 3. Used characters in distinct passwords

Character Classes The distribution and sequence of character classes used in passwords are visualized in Figure 3. Adding the three largest pieces in the diagram of Figure 3a shows that 81% of the available passwords only rely on lower letters and digits. For the sequences of character classes, 64% of all passwords are created with a sequence of any number of lower letters followed by any number of digits (`[a-z]*[0-9]*`).

Top Passwords Table 4 lists the 20 most used passwords of all leaks. For generating it, we took the average position of the passwords across all leak groups.

Table 4. Normalized top passwords

	Top 1-5		**Top 6-10**		**Top 11-15**		**Top 16-20**
1	123456	6	password	11	000000	16	abc123
2	111111	7	1q2w3e4r	12	1234567890	17	123qwe
3	12345678	8	1qaz2wsx	13	666666	18	654321
4	123456789	9	1234567	14	123321	19	112233
5	123123	10	iloveyou	15	qwerty	20	11111111

All of the 20 most used passwords can be categorized as weak passwords. They are either simple keyboard walks (`123456`, `1q2w3e4r`), obvious code words (`password`, `iloveyou`), repetitions of substrings (`123123`, 112233) or repetitions of single characters (000000, 666666). This ranking makes clear again that users use simple passwords, which are easier to remember, rather than ones that are more complex and harder to crack.

4.4 Password Reuse

One of the main attack vectors for recent cyber-attacks is the reuse of passwords across multiple accounts. This section takes a detailed look on the reuse of passwords in our dataset. To have a realistic view on the reuse, we will only take service database dumps for the analysis, because they contain independently chosen passwords. For credential collections, it cannot be ensured that the data was already obtained from reused passwords.

For our analysis, we check how often a user reuses the same email address with the same password as login credentials for different websites. As an additional info, we are also interested in how this reuse behaves across different types of services, for example dating sites.

We developed an algorithm, which analyzes the similarity of different passwords used by the same email address. For calculating the similarity measure between the passwords we used the normalized Levenshtein distance algorithm. This algorithm calculates the distance between two strings (S1, S2) by counting the operations on single-characters needed to convert S1 to S2. The algorithm differentiates between insertions, substitutions and deletions. The more similar two strings are, the less operations are needed for the conversion and the smaller is the resulting distance. The normalized similarity is calculated by subtracting the distance, divided by the length of the longest string, from 1 (1-(distance/longest string)).

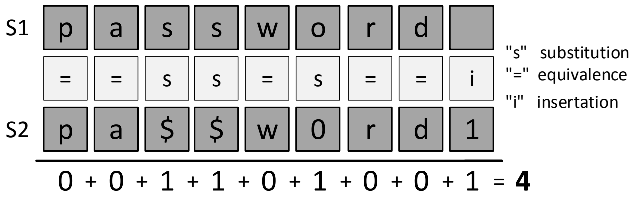


Fig. 4. Levenshtein distance algorithm

Figure 4 depicts the calculation of the similarity between two passwords based on the normalized Levenshtein algorithm. May a user use the passwords password (S1) and pa$$w0rd1 (S2) with the same email address for two different websites. For calculating the similarity between S1 and S2, the algorithm starts to compare character by character from left to right. So the first comparison would be between "p" and "p". Because they are the same character no operation is needed. The next comparison would be between "a" and "a". Again no operation is necessary. With the next comparison between "s" and "$" the algorithm detects a difference between the characters. For changing "s" to "$" a substitution is needed and the distance increases by 1. The next comparison is again between "s" and "$", so the distance increases to 2. The next comparison does not find any differences between the characters "w" and "w", but the next comparison detects a necessary substitution for changing "o" to "0". So the distance increases to 3. The next two characters are the same. After that, S1 ended, while S2 includes one more character, so the distance increases finally to 4. In conclusion, the normalized Levenshtein similarity between `password` and `pa$$w0rd1` is 0.56 (1 -(4/9)).

For every unique email address we built a weighted graph, which consists of the associated passwords as vertices and the similarity values between the vertices as weighted edges. After that, the graph is masked by only leaving the edges with a similarity value higher than 0.7. This keeps only the connection between passwords that are highly related and which indicate password reuse.

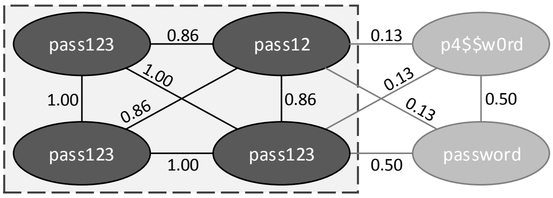


Fig. 5. Maximum clique of passwords

In the next step we used the masked subgraph for finding maximum cliques. A clique is a subset of all the nodes from an undirected graph that creates a new complete subgraph [20], a graph with edges between all vertices. A clique is called maximal, when none of the vertices is part of another clique. For finding all the maximal cliques we used the Bron-Kerbosch algorithm.

In our case, a maximal clique from one single email address consists of all the associated passwords, which have at least 70% similarity. All the other passwords were not considered. We determined that the minimal number of vertices within a clique shall not be less than 2. Furthermore, we calculated an average score for every clique by summing up all the scores from the edges and divide the result by the number of edges. So, if a clique has an average score of 1, all the passwords are exactly identical. This method allows us to analyze a large collection of database breaches in regards to the reuse of passwords with the same email address as login credentials for different websites.

As a result of our analysis we found 68.5 million email addresses that appear in more than one data breach. Within these email addresses, we could find $\approx$ 19 million email addresses (27%) with maximal cliques, which means they reuse passwords across websites with at least 70% similarity.

To find out the addresses that exactly reuse the same password, we set the minimum clique score to 1.0. In the end, we found about 13.7 million addresses (20%) with this property. Approximately 12.9 million of these addresses use exactly the same password for 2 websites, about 825.000 addresses use the same credentials for 3 websites and about 60.000 addresses use the same login data for 4 different websites.

In a last analysis, we created a matrix for password reuse among each two distinct service database dumps. The full matrix is printed in Appendix A for completeness. It shows the percentage of password reuse from all the email addresses that are included in both of the distinct sources. The highest rate of password reuse can be found between *Xiaomi* and *51CTO*. In both leaks there are 29,338 matching e-mail addresses and 70% of those addresses have the same password. Both pages are Chinese and have content on electronics. The lowest rate of password reuse can be found between the Chinese Software Developer Network *csdn.net* and the Russian social media network *sprashivai.ru*, because they do not have any email addresses in common. A reason could be different areas of interest, languages, and cultures. Apart from that, the reuse rate between other services is in the range of 0% to 70%.

4.5 Language Analysis

Another objective was to analyze the passwords from country-specific domains. We assumed that there will be typical terms from the respective language, which are used as passwords. As a result it should be possible to trace back the origin of a leak by analyzing its passwords.

We grouped the distinct email addresses by TLD and counted the amount of addresses for each domain. Table 5 shows the domains with the largest number of valid email addresses and passwords. Approximately 73% of all the distinct addresses belong to the generic TLD `.com`. Because of the international character of this domain we focused on more specific domains, i.e. `.nl`, `.cn`, `.ru`, `.fr`, `.it`, `.uk` and `.de` for further analyses.

In order to filter only the country-specific passwords for every domain we created a top 1000 list of the most used passwords from all the leaks. This list represents a general overview of passwords that are used all over the world. In the course of creating the top passwords list for every domain, we compared each password with the top 1000 list. If a password appears in this list, it is not country-specific enough. After filtering out these passwords, we ordered the domain-specific lists by frequency of occurrence and exported the 20 most used passwords.

Table 5. Country-specific passwords

ID	**Domain**	**Language**	**number of addresses**	**Top 5 passwords**
1	.uk	British English	18 604 736	liverpool, arsenal, chelsea
2	.fr	French	32 207 859	azerty, marseille, doudou
3	.de	German	15 401 823	passwort, ficken, qwertz
4	.it	Italian	21 856 935	juventus, andrea, francesco
5	.nl	Dutch	3 513 385	welkom, welkom01, wachtwoord
6	.cn	Chinese	12 213 153	5201314, woaini, 1314520
7	.ru	Russian	119 002 753	qwertyuiop, UsdopaA, 1q2w3e4r5t

Those top 20 lists illustrate that many users use words out of their native language as login credentials for websites. Good examples are the top passwords the United Kingdom, i.e. ".uk". The most commonly appearing password is "liverpool" - a major city in England with a famous, eponymous football club. On the second and third place follow the passwords "arsenal" and "chelsea", which are also names of famous football clubs in England. Other often used passwords belong to the British Royal Family ("william", "george") or other cities ("london", "manchester"). The top-level domain of France ends with ".fr". The top password of this domain is "azerty" - a walk on a keyboard with the AZERTY layout, which is typical for French keyboards. The password "marseille" reaches the second place and points to the famous city in France. Other passwords in the top 20 list of the French domain are "doudou" (soft toy), "loulou" (darling) and "chouchou" (little hearts). The top list of passwords from the German top-level domain ".de" is full of typical German first names like "annalena", "franzi", and "renate". Other country-specific first names can be found in the top list of Italy ("francesco", "giuseppe", "antonio").

In summary, the top-password lists for each country-specific domain include typical names and terms from the specific language. This fact can be used for hash cracking, because the usage of country-specific word lists for each domain would be more efficient for brute force attacks than universal dictionaries in case of employed secure hash functions.

5 Improving the Efficiency of Password Cracking

A good example for the risks of password reuse is the leak of the Ashley Madison database. Approximately half of the passwords in this data breach is encrypted by using the strong bcrypt function with a cost factor of 12. For this reason it is an extremely compute intensive task to decrypt these hashes.

Our approach was to check, whether there are email addresses included in the Ashley Madison leak, which appear in other leaks, too. When a user uses the same email address and the same associated password as login credentials for several other websites, it is most likely that the same password is used for Ashley Madison, too, given the email address matches. On the basis of this theory we collected all the credentials from other leaks, where the email address is included in the Ashley Madison data leak. After that we tried to decrypt the bcrypt hashes based on the cleartext passwords. As a result we were able to decrypt about 2.9 millions of bcrypt hashes within a few hours.

A particularly conspicuous aspect is the high matching rate of login credentials between similar websites. Ashley Madison and Mate1 can be categorized as dating platforms. We compared the included credentials in both leaks and found 1.94 millions of matching email addresses. Then used the cleartext passwords from Mate1 for decrypting the bcrypt hashes from Ashley Madison. With our hardware[8] it took approximately 5 hours for hashing all the cleartext passwords and then comparing them with the hashes. Within these 5 hours we were able to decrypt 913 550 bcrypt hashes. In comparison with this procedure, the decryption of these hashes using the top 20 passwords as a word list only found 1800 passwords within 5 hours.

Furthermore, about 42% of the users that used the same email address for both websites, used the same password, too. Conversely, between Ashley Madison and MySpace there are 4.54 million matching email addresses and only 1.1 million matching passwords (24%). This could indicate that the rate of password reuse for similar websites is higher than the rate for websites with different content.

[8] Virtual machine with 54 cores (Intel® Xeon® Processor X7560 with 2.27 GHz), we used a powerful CPU over a high-end GPU, as bcrypt is known to perform poor with GPUs

6 Conclusion and Future Work

In this paper, we wanted to analyze the selection of passwords of Internet users based on a huge dataset of real-world credentials from 31 publicly identity leaks. Our main focus in this analysis was the reuse of passwords and the regional use of passwords.

For the regional use of passwords, we could find that users mainly take their passwords from popular first names and cities in a country. Additionally, there are certain terms around the topic of love that are specific to the language of the users' country. It shows that passwords can indeed be specific to the origin of users and therefore the origin of a leak. Thus, the regional password selection is a possible way to derive the origin of a whole leak.

For the issue of password reuse, we were able to find around 68.5 million Internet users that have credentials on multiple leaked websites. Among these, 13.7 million (20%) reused exactly the same password on 2 different websites. Even more, i.e. 18.9 million users (27%), use a similar or equal password (>70% similarity). Interestingly, the password reuse for the matching users is highly dependent on the similarity of the services. For similar services, users often use the same password. We found that some sites even have a password reuse rates of up to 70%, while some sites have reuse rates as low as a few percent.

As a demonstration for the problem of password reuse, we had a closer look at the Ashley Madison-leak. The leak is known to be resistant to cracking attacks because of its strong bcrypt hash function. Looking up credentials of Ashley Madison users from various sources, we were able to recover around 2.9 million passwords in a few hours. To crack this amount with traditional means would have probably taken multiple month.

For future work, there should be more analysis on the reuse patterns and rates between similar platforms, as we expect that users more likely reuse their passwords over services in similar domains.

Another possible topic could be the password similarity in the area of password reuse. Our performed analysis showed that users add small changes to their passwords for creating *new* passwords. Nevertheless, the similarity between these passwords is often higher than 70%. It can be analyzed, which routines users use to manipulate their passwords. This could help to warn the operators in case of password changes that the new password is too similar to the old one.

References

[1] Joseph Cox. *Another Day, Another Hack: 100 Million Accounts for VK, Russia's Facebook.* Web site. June 2016. URL: http://motherboard.vice.com/read/another-day-another-hack-100-million-accounts-for-vk-russias-facebook (visited on 07/01/2016).

[2] Lorenzo Franceschi-Bicchierai. "Another Day, Another Hack: 117 Million LinkedIn Emails And Passwords". In: *Vice Motherboard* (May 2016). URL: http://motherboard.vice.com/read/another-day-another-hack-117-million-linkedin-emails-and-password (visited on 07/01/2016).

[3] Lorenzo Franceschi-Bicchierai. "Hacker Tries To Sell 427 Milllion Stolen MySpace Passwords For $2,800". In: *Vice Motherboard* (May 2016). URL: http://motherboard.vice.com/read/427-million-myspace-passwords-emails-data-breach (visited on 07/01/2016).

[4] Lorenzo Franceschi-Bicchierai. "Hackers Stole 65 Million Passwords From Tumblr, New Analysis Reveals". In: *Vice Motherboard* (May 2016). URL: https://motherboard.vice.com/read/hackers-stole-68-million-passwords-from-tumblr-new-analysis-reveals (visited on 07/01/2016).

[5] Joseph Cox. "Your Shitty Password Hygiene Is Spreading Hacks Like a Contagion". In: *Vice Motherboard* (June 2016). URL: http://motherboard.vice.com/read/your-shitty-password-hygiene-is-spreading-hacks-like-a-contagion-twitter-logins-hacked (visited on 07/01/2016).

[6] Alex Hern. "Mark Zuckerberg hacked on Twitter and Pinterest". In: *The Guardian* (June 2016). URL: https://www.theguardian.com/technology/2016/jun/06/mark-zuckerberg-hacked-on-twitter-and-pinterest (visited on 07/01/2016).

[7] HAL 90210. "Oculus CEO is latest tech boss hacked in embarrassing account takeover". In: *The Guardian* (June 2016). URL: https://www.theguardian.com/technology/2016/jun/30/oculus-ceo-is-latest-tech-boss-hacked-in-embarrassing-account-takeover (visited on 07/01/2016).

[8] Verizon. *2016 Data Breach Investigations Report.* Apr. 2016. URL: http://www.verizonenterprise.com/verizon-insights-lab/dbir/2016/ (visited on 07/01/2016).

[9] CSID. *Consumer Survey: Password Habits. A study of password habits among American consumers.* Tech. rep. CSID, 2012. URL: https://www.csid.com/wp-content/uploads/2012/09/CS_PasswordSurvey_FullReport_FINAL.pdf.

[10] David Jaeger, Hendrik Graupner, et al. "Gathering and Analyzing Identity Leaks for Security Awareness". In: *Proceedings of the 7th International Conference on Passwords (PASSWORDS'14).* Vol. 9393. Lecture Notes in Computer Science (LNCS). Trondheim, Norway: Springer International Publishing, Dec. 2014, pp. 102–115. ISBN: 978-3-319-24191-3. DOI: 10.1007/978-3-319-24192-0_7.

[11] Hendrik Graupner, David Jaeger, et al. "Automated Parsing and Interpretation of Identity Leaks". In: *Proceedings of the 13th Computing Frontiers Conference 2016 (CF'16).* Como, Italy: ACM, May 2016, pp. 127–134. ISBN: 978-1-4503-4128-8. DOI: 10.1145/2903150.2903156.

[12] Anupam Das, Joseph Bonneau, et al. "The Tangled Web of Password Reuse". In: *21nd Annual Network and Distributed System Security Symposium (NDSS'14).* Feb. 2014. DOI: 10.14722/ndss.2014.23357.

[13] Recorded Future. *Government Credentials on the Open Web.* Tech. rep. Recorded Future, 2015. URL: http://media.scmagazine.com/documents/131/recorded_future_32569.pdf.

[14] Dennis Mirante and Justin Cappos. *Understanding Password Database Compromises.* Tech. rep. Technical Report TR-CSE-2013-02, Department of Computer Science and Engineering Polytechnic Institute of NYU, 2013.

[15] Matteo Dell'Amico, Pietro Michiardi, and Yves Roudier. "Password strength: An empirical analysis". In: *Proceedings of the 29th Conference on Computer Communications (INFOCOM'10).* IEEE, 2010.

[16] Troy Hunt. *A brief Sony password analysis.* Blog entry. June 2011. URL: https://www.troyhunt.com/brief-sony-password-analysis/ (visited on 06/22/2016).

[17] CynoSure. *How we cracked millions of Ashley Madison bcrypt hashes efficiently*. Blog. Sept. 2015. URL: http://cynosureprime.blogspot.de/2015/09/how-we-cracked-millions-of-ashley.html (visited on 07/12/2016).

[18] Mohit Kumar. “Tianya, China’s biggest online forum 40 million users data Leaked”. In: *The Hacker News* (2011). URL: http://thehackernews.com/2011/12/tianya-chinas-biggest-online-forum-40.html (visited on 07/01/2016).

[19] Jens Steube. “PRINCE: Modern Password Guessing Algorithm”. In: *International Conference on Passwords (PASSWORDS’14)*. Dec. 2014.

[20] Karin Haenelt. *Cliquen in Graphen: Mathematische Grundlagen und der Bron-Kerbosch-Algorithmus*. 2012. URL: http://kontext.fraunhofer.de/haenelt/kurs/folien/Haenelt_Clique.pdf.

A Password Reuse Matrix

ID	Source	ID	Source
1	000webhost.com	12	mpgh.net
2	17.media	13	myspace.com
3	51cto.com	14	naughtyamerica.com
4	aipai.com	15	nexusmods.com
5	ashleymadison.com	16	r2games.com
6	badoo.com	17	sprashivai.ru
7	csdn.net	18	tianya.cn
8	gawker.com	19	vk.com
9	imesh.com	20	xiaomi.com
10	linkedin.com	21	xsplit.com
11	mate1.com		

1	-																				
2	22.1	-																			
3	18.7	44.5	-																		
4	23.6	39.2	57.1	-																	
5	18.9	3.8	21.3	22.0	-																
6	7.1	10.0	23.6	17.5	22.4	-															
7	22.8	23.1	38.9	39.4	17.2	14.3	-														
8	13.1	13.5	42.9	28.6	34.0	15.2	22.1	-													
9	14.2	18.0	30.0	23.9	22.9	15.7	26.0	37.9	-												
10	20.6	33.4	58.6	53.8	28.6	15.4	33.3	15.2	38.4	-											
11	13.5	10.9	18.4	15.6	42.0	20.0	17.5	31.8	32.3	32.0	-										
12	14.7	17.3	24.4	30.3	25.6	7.3	20.3	22.7	26.1	24.2	15.9	-									
13	17.7	13.3	23.5	23.9	19.0	8.4	17.7	16.7	18.2	22.4	16.7	14.0	-								
14	21.0	26.4	20.8	36.0	45.7	19.4	24.5	35.0	41.7	41.3	40.6	27.4	22.4	-							
15	26.9	45.1	61.4	49.4	21.0	14.4	34.3	33.5	41.9	40.1	35.2	28.1	20.4	42.6	-						
16	20.1	19.7	21.9	33.7	33.7	6.2	23.4	10.1	20.9	18.8	16.7	25.2	11.6	31.7	44.2	-					
17	19.7	9.6	2.5	3.8	5.6	6.6	0.0	7.6	24.8	22.9	12.4	19.9	15.2	25.3	42.0	26.4	-				
18	14.7	33.3	61.6	51.0	20.3	17.7	33.6	37.5	18.1	46.5	14.3	13.1	14.6	26.4	52.3	11.6	6.0	-			
19	17.6	14.0	32.9	33.6	22.7	5.2	26.4	27.5	29.4	31.6	24.2	16.0	13.7	31.4	23.9	14.6	12.1	29.2	-		
20	22.7	48.1	70.0	58.9	20.8	25.4	36.3	40.2	39.5	64.3	23.6	30.5	28.5	38.2	59.9	30.8	34.7	59.4	40.6	-	
21	37.8	49.9	56.1	52.0	48.0	13.7	36.8	25.1	36.9	43.7	37.1	24.2	23.0	46.8	52.5	52.8	49.2	41.0	21.5	60.2	-
ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

Table 6. Password reuse (in percent)