Lost in Disclosure: On The Inference of Password Composition Policies

Conference Paper · September 2019

DOI: 10.1109/ISSREW.2019.00082

4 authors:

Saul Johnson
Teesside University
4 PUBLICATIONS **2** CITATIONS
SEE PROFILE

João F. Ferreira
Instituto Superior Técnico
34 PUBLICATIONS **153** CITATIONS
SEE PROFILE

Alexandra Mendes
Universidade da Beira Interior
18 PUBLICATIONS **57** CITATIONS
SEE PROFILE

Julien Cordry
Teesside University
20 PUBLICATIONS **25** CITATIONS
SEE PROFILE

Lost in Disclosure: On The Inference of Password Composition Policies

Saul Johnson*, João F. Ferreira†, Alexandra Mendes‡ and Julien Cordry*
*Software and Systems Research Group, Teesside University, Middlesbrough, UK
†Instituto Superior Técnico, University of Lisbon and INESC-ID, Lisbon, Portugal
‡HASLab, INESC TEC and Computer Science Department, University of Beira Interior, Covilhã, Portugal
Email: *{saul.johnson,j.cordry}@tees.ac.uk, †joao@joaoff.com, ‡alexandra@archimendes.com

***Abstract*—Large-scale password data breaches are becoming increasingly commonplace, which has enabled researchers to produce a substantial body of password security research utilising real-world password datasets, which often contain numbers of records in the tens or even hundreds of millions. While much study has been conducted on how password composition policies—sets of rules that a user must abide by when creating a password—influence the distribution of user-chosen passwords on a system, much less research has been done on inferring the password composition policy that a given set of user-chosen passwords was created under. In this paper, we state the problem with the naive approach to this challenge, and suggest a simple approach that produces more reliable results. We also present *pol-infer*, a tool that implements this approach, and demonstrates its use in inferring password composition policies.**

***Index Terms*—password composition policy, security, inference, big data**

I. Introduction

When cybercriminals compromise a user credential database and release its contents into the public arena, a number of different interested parties might seek to obtain and use the data it contains, with varying goals in mind. These might include, for instance, other groups of cybercriminals seeking to employ the data in credential stuffing attacks [1], and security researchers seeking to understand user password choice on the system concerned [2]–[4]. In particular, the latter group may be concerned with the *password composition policy* the passwords in the database were created under, in order to better understand how these rules around user password creation affect the distribution of user password choices.

Security researchers may find themselves confounded in this endeavour, however, because when the breached user credential database is released to the public, information about the password composition policy in place at the time of the breach is often not included. This could be because the party behind the breach does not think it relevant, wishes to keep their methods as secret as possible, or never sought this information out in the first place—after all, the password composition policy is of comparatively little interest to malicious actors seeking to directly employ the credentials in the database to criminal ends. The only other party known to have this information is the organisation that was the victim of the data breach in the first place, who by this point may be unable or unwilling to disclose any information regarding their security practices. Reasons for this might include, for example:

- The organisation may have ceased to exist entirely, prior to the time at which the research in question is being conducted. There are several examples of this happening in the real world, for example the now-defunct Christian dating site *singles.org* [5] which ceased to exist sometime after 2009 when their entire user credential database was compromised in plaintext.
- The organisation might be understandably reluctant to disclose any information regarding their security practices for fear of being further targeted or incriminating themselves by confessing to having taken inadequate measures to safeguard user data. This is especially the case in Europe, where tightening legislation around data protection [6] might make the latter point of particular concern.

If we cannot obtain a description of the password composition policy from any of the organisations involved in the breach, this information has been *lost in disclosure*—that is, lost somewhere in the process of the transfer of data between parties. We are therefore forced to turn to the data that we do have to attempt to infer as much of that lost information as we can.

TABLE I
The four real-world breached password datasets studied in this work, alongside their corresponding policies according to [7], [8], and numbers of passwords within them.

Dataset	Policy	Size
RockYou [9]	$length \geq 5$	32,603,048
Yahoo [10]	$length \geq 6$	453,492
000webhost [11]	$length \geq 6 \wedge digits \geq 1$	15,271,208
LinkedIn [12]	$length \geq 6$	172,428,238

There is no shortage of breached user credential databases available online. Arguably the most well-known of these, the RockYou set [9], like many others (e.g. the Yahoo [10] or 000webhost [11] sets) contains passwords that do not comply with the password composition policy in place when the breach happened (see Tables I and II). Reasons for this "noise" vary, but include:

- Multiple password composition policies per dump—the RockYou set, for example, is an aggregate made up of at

least two tables: one containing passwords to the main web application and one containing passwords used to log in to "partner services" (e.g. MySpace) which may enforce different policies [9]. Passwords created under old policies may also be present. RockYou, for instance, changed their policy after their data breach in 2009 from minimum 5 characters in length [7] to a stronger policy [8], [13]. In this case, our methodology gives the password composition policy that the majority of passwords were created under, though there is scope for improving upon this in future work (see Section VI).

- Formatting errors—when the raw data is being processed by the exfiltrating party, errors may be introduced if their data processing scripts are not robust. For example, passwords containing spaces may be read as two separate data points.
- Intentional padding—if cybercriminals initially offer the data for sale, the price that they are capable of obtaining is often contingent on the number of records it contains. It is therefore possible that the dataset may be intentionally padded with extra records, some of which might contain non-compliant passwords.

TABLE II
A BREAKDOWN OF THE NUMBER OF COMPLIANT AND NON-COMPLIANT PASSWORDS PRESENT IN EACH DATASET LISTED IN TABLE I, ACCORDING TO [7], [8].

Dataset	Compliant	Non-compliant
RockYou [9]	32524461	78587 (0.24%)
Yahoo [10]	444942	8550 (1.89%)
000webhost [11]	14936872	334336 (2.19%)
LinkedIn [12]	172409689	18549 (0.01%)

With "noisy" data like this, we cannot, for example, simply check for the shortest password in the database to determine the minimum password length constraint specified by the policy. In fact, the authors of one published work [14] mention in their publication that the presence of "non-password artifacts" in the RockYou dataset factored in to their choice of research methods, at least in part due to the difficulty of filtering these out. This motivates us to search for a simple, easy-to-implement method to attempt to infer password composition policy rules from a password dataset, which would make filtering out at least some of these artifacts trivial. The remainder of this work outlines an alternative approach that we have found success with.

a) Contribution: We make the following concrete contributions in this work: (i) for the first time, we draw attention to the problem of "noise" in publicly-available breached password datasets in the form of passwords that do not comply with the password composition policy in place when the breach occurred (ii) we suggest an easy-to-implement approach to filtering out this noise by converting the problem to one of outlier detection, without consulting any organisation involved in the breach (iii) we make *pol-infer* [15] available[1], the tool used to produce the data and visualisations in our results (Section IV and Section V).

b) Outline: We have introduced and motivated the work in this Section I. We describe related work in Section II. In Section III we describe our approach in detail, showing the results we are able to obtain from the four password datasets shown in Table II in Section IV. In Section V we apply our methodology to datasets created to simulate both intentional padding and processing with error-prone data processing scripts. We conclude in Section VI, discussing the limitations of our approach and potential future work.

II. RELATED WORK

We are not aware of any existing published work that explores the automation of password composition policy inference from large datasets. Previous research has involved determining the password composition policies used by active services. A study by Florêncio and Herley [13] gathered password composition policy information by creating an account on the service, where possible, and performing web searches otherwise. This study was later replicated by Mayer et al. in [8]. In [7], Golla and Dürmuth make extensive use of password data dumps where the password composition policy is known.

III. METHODOLOGY

Our approach is applicable to any numerically-typed password attribute α which is a function of type $Password \to \mathbb{N}$ which extracts some password property (e.g. length). By default, *pol-infer* supports the password attributes in Table III, sufficient to capture the policies used in the study by Shay et al. [16] with the exception of the dictionary check on the *comprehensive8* policy, which cannot be expressed as an attribute of this type.

TABLE III
PASSWORD ATTRIBUTES USABLE WITH *pol-infer* BY DEFAULT. ANY ATTRIBUTE APPEARING THE TABLE BELOW CAN BE USED BY THE TOOL TO INFER PASSWORD COMPOSITION POLICIES.

Attribute (α)	Description
length	The number of characters in the password (i.e. its length).
words	The number of words in the password. We define "words" in the same was as in [16]—as "letter sequences separated by a nonletter sequence".
lowers	The number of lowercase letters in the password.
uppers	The number of uppercase letters in the password.
digits	The number of digits in the password.
symbols	The number of non-alphanumeric characters in the password.
classes	The number of character classes in the password. We recognise four character classes in the popular LUDS scheme—lowercase, uppercase, digits and symbols.

For instance, let us suppose we wish to infer the minimum length constraint specified by the policy that the 000webhost set [11] was created under (that is, $\alpha = length$). In this case, previous research [7] has established that the answer is 6, and

[1]Available for download at: https://sr-lab.github.io/pol-infer/

yet the data in Table IV would seem to contradict this—there are passwords shorter than this present in the data.

TABLE IV
FREQUENCIES $f(l)$ OF PASSWORDS OF DIFFERENT LENGTHS l IN THE 000WEBHOST SET [11], ALONGSIDE THEIR CUMULATIVE FREQUENCIES $cum(l)$ AND THE MULTIPLIER $mult(l)$ REQUIRED TO REACH THE CUMULATIVE FREQUENCY OF THE NEXT LENGTH $cum(l+1)$.

l	$f(l)$	$cum(l)$	$mult(l)$
1	306	306	6.03
2	1540	1846	1.42
3	775	2621	1.47
4	1221	3842	1.66
5	2456	6388	137.23
6	870209	876597	2.38
7	1208092	2084689	—

It is readily apparent how the data in Table IV may be used to determine the minimum length constraint in the 000webhost policy. By observing the outlying value of 137.23 in the $mult(l)$ column, we can see that we now have an outlier detection problem. In Table IV, for every length l:

$$mult(l) = \frac{cum(l+1)}{cum(l)}$$

We can infer the minimum password length enforced by the password composition policy under which this data was created by looking for the outlying "sudden increase" in $f(l)$, taking $l+1$ where:

$$mult(l) = max(\{mult(m)|m \in \mathbb{N}\})$$

For the 000webhost data, this gives us the correct answer 6. By examining the number of digits in a password, as opposed to password length (that is to say $\alpha = digits$), we are also able to determine that the 000webhost policy demands that passwords contain at least one digit (see Section IV).

By setting a lower threshold on $mult(\alpha)$ we are able to specify a cutoff point c below which we assume there is no constraint in place on the attribute α in question. For $\alpha \in \{length, digits, uppers\}$, we have found success using a value of 2 as this threshold (i.e. $c = 2$). For example, consider that the 000webhost policy does not demand that any uppercase letters be present in passwords.

TABLE V
FREQUENCIES $f(u)$, CUMULATIVE FREQUENCIES $cum(u)$ AND MULTIPLIERS $mult(u)$ OF PASSWORDS CONTAINING DIFFERENT NUMBERS OF UPPERCASE LETTERS u IN THE 000WEBHOST SET [11].

u	$f(u)$	$cum(u)$	$mult(u)$
0	12366006	12366006	1.08
1	1049727	13415733	1.02
2	315637	13731370	1.02
3	267042	13998412	1.02
4	260061	14258473	1.02
5	241305	14499778	1.02
6	220202	14719980	1.01
7	187806	14907786	—

As no value in Table V is outlying above the default cutoff point of 2, we conclude that there was likely no constraint on minimum number of uppercase letters present in the password policy when the dataset was created.

IV. RESULTS: REAL DATA

We present a set of results demonstrating the success of our approach when used to infer minimum password length specified by the policy under which 4 different data sets were created.

- **RockYou**—breached in plaintext from an online gaming service of the same name circa 2009 [9]. The policy in place at the time enforced a minimum length of 5 characters, with no other constraints [7]. Contains a total of 32,603,048 passwords.
- **Yahoo**—breached from the Yahoo Voice VoIP service circa 2012 [10]. The policy in place at the time of the breach enforced a minimum length of 6 characters with no other requirements [8]. Contains 453,492 passwords.
- **000webhost**—breached from the web hosting service of the same name circa 2015 [11]. The policy in place at the time of the breach enforced a minimum length of 6 characters, with at least one numeric digit [7]. Contains 15,271,208 passwords.
- **LinkedIn**—breached from the professional social networking site of the same name circa 2012, the true extent of this breach was uncovered in 2016 as much bigger than was initially made public [12]. Unsalted password hashes in SHA-1 format were extracted, of which $\approx 98\%$ have since been cracked. It is these cracked passwords we use in this work. The policy in place at the time of the breach enforced a minimum length of 6 characters with no other requirements [8]. Contains 172,428,238 passwords.

The results that follow were produced using *pol-infer*—a tool we make available [15] for inferring password composition policies from large datasets using the approach we describe in Section III.

A. The RockYou Set (2009)

Previous research has established that the majority of the RockYou set [9] was created under a policy enforcing minimum length 5 with no other requirements [7].

The outlying point at $l = 4$ in Figure 1 indicates that the password composition policy that most of the passwords in the set were created under enforces a minimum length of 5. This aligns with existing literature [7].

B. The Yahoo Set (2012)

Previous research has established that the majority of the Yahoo set [10] was created under a policy enforcing minimum length 6 with no other requirements [8].

The outlying point at $l = 5$ in Figure 2 indicates that the password composition policy that most of the passwords in the set were created under enforces a minimum length of 6. This aligns with existing literature [8].

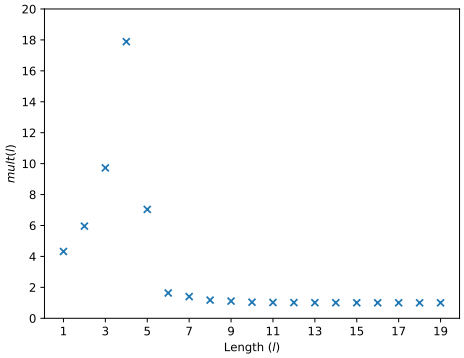

Fig. 1. Passwords of different lengths l in the RockYou set [9], plotted against $mult(l)$.

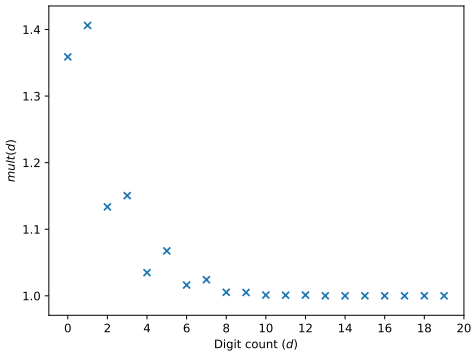

Fig. 3. Passwords containing different numbers of digits d in the Yahoo set [10], plotted against $mult(d)$.

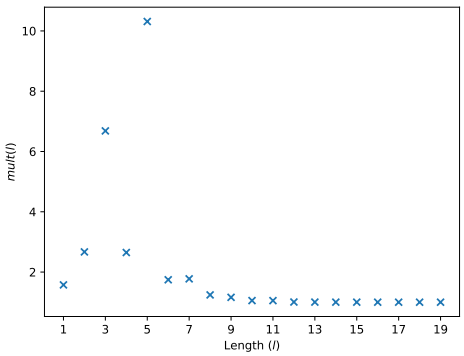

Fig. 2. Passwords of different lengths l in the Yahoo set [10], plotted against $mult(l)$.

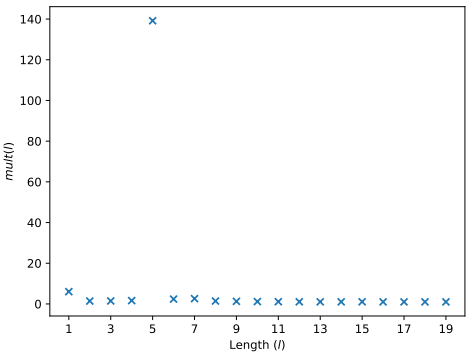

Fig. 4. Passwords of different lengths l in the 000webhost set [11], plotted against $mult(l)$.

1) Inferring the Absence of Constraints: As no points in Figure 3 are present above the default *pol-infer* [15] cutoff point of $c = 2$, the tool indicates that there was likely no constraint on minimum number of digits present in the password policy when the Yahoo dataset was created. This aligns with existing literature [8].

C. The 000webhost Set (2015)

Previous research has established that the majority of the 000webhost set [11] was created under a policy enforcing minimum length 6 with the additional requirement that passwords must contain at least one digit [7].

The outlying point at $l = 5$ in Figure 4 indicates that the password composition policy that most of the passwords in the set were created under enforces a minimum length of 6. This aligns with existing literature [7].

The outlying point at $d = 0$ in Figure 5 indicates that the password composition policy that most of the passwords in the set were created under enforces a minimum of 1 digit in passwords.

D. The LinkedIn Set (2016)

Previous research has established that the majority of the LinkedIn set [12] was created under a policy enforcing minimum length 6 with no other requirements [7].

The outlying point at $l = 5$ in Figure 6 indicates that the password composition policy that most of the passwords in the set were created under enforces a minimum length of 6. This aligns with existing literature [7].

V. Results: Synthetic Data

In order to simulate the effect of some of the circumstances mentioned in Section I that could potentially create non-compliant "noise" in real-world password datasets, we created the following synthetic datasets:

- **2word12_linkedin_padded**—The LinkedIn dataset [12] filtered according to a 2word12 policy (at least 12 characters long, at least 2 letter sequences separated by a non-letter sequence) to leave 1,511,786 passwords. This has then been combined with the *singles.org* dataset [5] (16,248 passwords), *elitehacker* dataset (1000 passwords), *hak5* dataset [17] (2987 passwords), and *faithwriters* dataset [18] (9709 passwords). This is designed

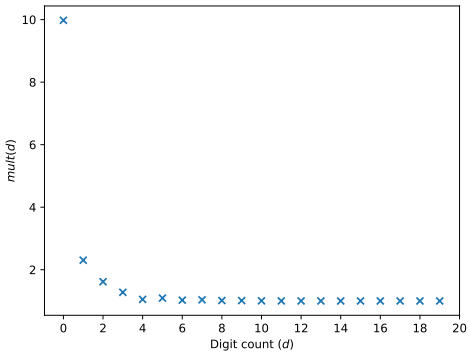

Fig. 5. Passwords containing different numbers of digits d in the 000webhost set [11], plotted against $mult(d)$.

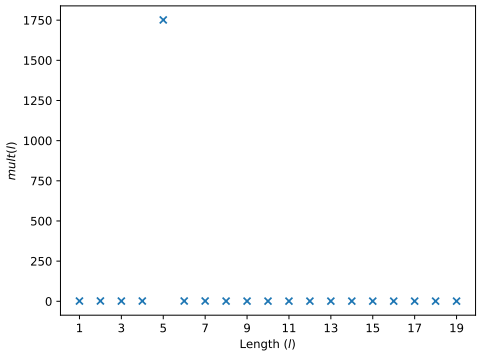

Fig. 6. Passwords of different lengths l in the LinkedIn set [12], plotted against $mult(l)$.

to simulate intentional padding of a dataset created under one policy with several other smaller datasets in order to increase its resale value.

- **2class8_linkedin_errors**—The LinkedIn dataset [12] filtered according to 2class8 policy (at least 8 characters long, at least 2 character classes present from lowercase, uppercase, digits and symbols) to leave 65,271,156 passwords. For every password in this dataset containing either a space or a comma, this password has then been split into two or more separate strings along these tokens, leading to the creation of 404,547 additional records. This simulates the type of formatting error that might be introduced by processing scripts after the dataset has been exfiltrated.

A. Intentional Padding

Figure 7 and Table VI show the use of our methodology to recover the original password composition policy of 2word12_linkedin_padded (2word12). The outlying points at $l = 11$ and $w = 1$ give us a length and word count of 12 and 2 respectively.

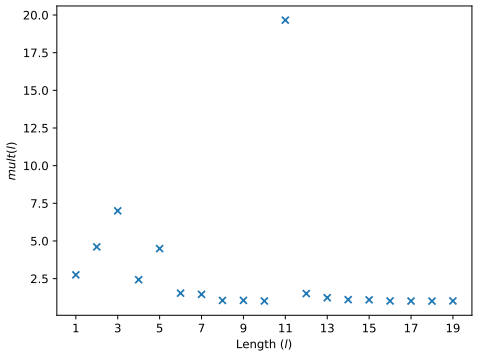

Fig. 7. Passwords of different lengths l in the 2word12_linkedin_padded synthetic dataset, plotted against the multiplier $mult(l)$ required to reach the cumulative frequency of the next length $cum(l + 1)$.

TABLE VI
FREQUENCIES $f(w)$, CUMULATIVE FREQUENCIES $cum(w)$ AND MULTIPLIERS $mult(w)$ OF PASSWORDS CONTAINING DIFFERENT NUMBERS OF WORDS w IN THE 2WORD12_LINKEDIN_PADDED SYNTHETIC DATASET.

w	$f(w)$	$cum(w)$	$mult(w)$
0	2500	2500	11.18
1	25460	27960	39.39
2	1073513	1101473	1.17
3	190996	1292469	1.07
4	89916	1382385	—

B. Formatting Errors

Figure 8 and Table VII show the use of our methodology to recover the original password composition policy of 2class8_linkedin_errors (2class8). The outlying points at $l = 7$ and $c = 1$ give us a length and class count of 8 and 2 respectively.

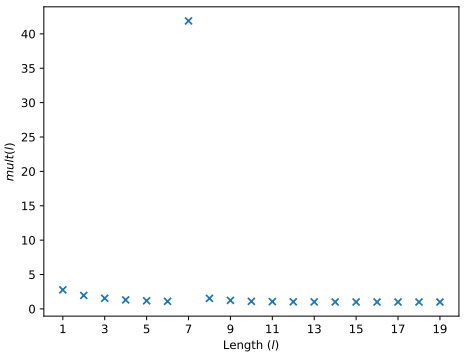

Fig. 8. Passwords of different lengths l in the 2word12_linkedin_errors synthetic dataset, plotted against $mult(l)$.

VI. CONCLUSION

In this work, we have demonstrated a simple, easy-to-implement methodology for inferring the password composition policy under which a password data dump was created

TABLE VII
FREQUENCIES $f(c)$, CUMULATIVE FREQUENCIES $cum(c)$ AND MULTIPLIERS $mult(c)$ OF PASSWORDS CONTAINING DIFFERENT NUMBERS OF WORDS c IN THE 2WORD12_LINKEDIN_ERRORS SYNTHETIC DATASET.

c	$f(c)$	$cum(c)$	$mult(c)$
1	591820	591820	84.87
2	49637360	50229180	1.27
3	13401629	63630809	1.03
4	2044894	65675703	—

without the need to interact with any of the parties involved in its disclosure. Once we have done this, we are able to trivially filter out non-compliant passwords if we so wish. We make *pol-infer*, the tool implementing this methodology that we used to produce the results in Sections IV and V, freely available [15]. We show that results obtained by this tool agree with existing literature on several real-world password datasets, and that it is effective on datasets generated to mimic those that might arise as a result of intentional padding or buggy data processing.

a) Limitations: While our approach is capable of approximately inferring password composition policies that place constraints on specific password attributes, it cannot offer a guarantee that the inferred policy is accurate or complete. As an example of a password composition policy rule that would be very difficult to infer, consider a rule that limits password length to a maximum of 1024 characters. As very few user-chosen passwords would be in violation of this rule even in its absence, its impact on user password choice would be very limited, making its inference very difficult.

b) Future work: Where time and date of account creation is available in password data dumps, it may be possible to detect with some accuracy the date and time of any password composition policy changes, offering new insight into the organisation's internal security practices. This may require *pol-infer* to become more modular, acting as a framework capable of hosting different inference algorithms. Work on *pol-infer* is planned to make policy inference more automated and comprehensive (e.g. inference of dictionary checks), with an option to generate password composition policy names in the style used by [16]. We plan to make use of *pol-infer* and the methodology we propose in this work to help prepare password data for use in research into other aspects of password security, such as formally verified password composition policy enforcement software [19].

REFERENCES

[1] M. Alsaleh, M. Mannan, and P. C. van Oorschot, "Revisiting defenses against large-scale online password guessing attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 1, pp. 128–141, Jan 2012.

[2] M. Weir, S. Aggarwal, M. Collins, and H. Stern, "Testing metrics for password creation policies by attacking large sets of revealed passwords," in *Proceedings of the 17th ACM Conference on Computer and Communications Security*, ser. CCS '10. New York, NY, USA: ACM, 2010, pp. 162–175.

[3] 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 *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, ser. CCS '13. New York, NY, USA: ACM, 2013, pp. 173–186.

[4] B. Ur, J. Bees, S. M. Segreti, L. Bauer, N. Christin, and L. F. Cranor, "Do users' perceptions of password security match reality?" in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ser. CHI '16. New York, NY, USA: ACM, 2016, pp. 3748–3760.

[5] D. Pauli, "Exposed web site a reminder for use of multiple passwords — network world," https://www.networkworld.com/article/2263760/exposed-web-site-a-reminder-for-use-of-multiple-passwords.html, Feb 2009, (Accessed on 07/25/2019).

[6] European Parliament, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)," *Official Journal of the European Union*, vol. 59, pp. 1–88, 2016.

[7] M. Golla and M. Dürmuth, "On the accuracy of password strength meters," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18. New York, NY, USA: ACM, 2018, pp. 1567–1582.

[8] P. Mayer, J. Kirchner, and M. Volkamer, "A second look at password composition policies in the wild: Comparing samples from 2010 and 2016," in *Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017)*. Santa Clara, CA: USENIX Association, 2017, pp. 13–28.

[9] N. Cubrilovich, "Rockyou hack: From bad to worse — techcrunch," https://techcrunch.com/2009/12/14/rockyou-hack-security-myspace-facebook-passwords/, Dec 2009, (Accessed on 04/10/2019).

[10] D. Gross, "Yahoo hacked, 450,000 passwords posted online - cnn," https://edition.cnn.com/2012/07/12/tech/web/yahoo-users-hacked, Jul 2012, (Accessed on 04/10/2019).

[11] C. Osborne, "000webhost hacked, 13 million customers exposed — zdnet," https://www.zdnet.com/article/000webhost-hacked-13-million-customers-exposed/, Oct 2015, (Accessed on 04/10/2019).

[12] M. Burgess, "Check if your linkedin account was hacked — wired uk," https://www.wired.co.uk/article/linkedin-data-breach-find-out-included, May 2016, (Accessed on 07/26/2019).

[13] D. Florêncio and C. Herley, "Where do security policies come from?" in *Proceedings of the Sixth Symposium on Usable Privacy and Security*, ser. SOUPS '10. New York, NY, USA: ACM, 2010, pp. 10:1–10:14.

[14] 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 *2012 IEEE Symposium on Security and Privacy*, May 2012, pp. 523–537.

[15] S. Johnson, "sr-lab/pol-infer: Inferring password composition policies from breached user credential databases." https://github.com/sr-lab/pol-infer, 4 2019, (Accessed on 04/12/2019).

[16] R. Shay, S. Komanduri, A. L. Durity, P. S. Huh, M. L. Mazurek, S. M. Segreti, B. Ur, L. Bauer, N. Christin, and L. F. Cranor, "Designing password policies for strength and usability," *ACM Trans. Inf. Syst. Secur.*, vol. 18, no. 4, pp. 13:1–13:34, May 2016.

[17] L. Constantin, "Security gurus 0wned by black hats," https://news.softpedia.com/news/Security-Gurus-0wned-by-Black-Hats-117934.shtml, Jul 2009, (Accessed on 05/10/2019).

[18] A. Greenberg, "Researcher creates clearinghouse of 14 million hacked passwords," https://www.forbes.com/sites/andygreenberg/2010/08/26/researcher-creates-clearinghouse-of-14-million-hacked-passwords/#7bacb64318fd, Aug 2010, (Accessed on 05/10/2019).

[19] J. F. Ferreira, S. A. Johnson, A. Mendes, and P. J. Brooke, "Certified password quality - A case study using Coq and Linux pluggable authentication modules," in *Integrated Formal Methods - 13th International Conference, IFM 2017, Turin, Italy, September 20-22, 2017, Proceedings*, 2017, pp. 407–421.