Journal of Information Security and Applications 84 (2024) 103809

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

A comprehensive evaluation on the benefits of context based password cracking for digital forensics

Aikaterini Kanta [a,b], Iwen Coisel [c], Mark Scanlon [a,*]

[a] Forensics and Security Research Group, School of Computer Science, University College Dublin, Ireland
[b] University of Portsmouth, Portsmouth, United Kingdom
[c] Europol, The Hague, The Netherlands

ARTICLE INFO

Keywords:
Password
Dictionary
Contextual information
Password cracking
Wordlist

ABSTRACT

Password-based authentication systems have many weaknesses, yet they remain overwhelmingly used and their announced disappearance is still undated. The system admin overcomes the imperfection by skilfully enforcing a strong password policy and sane password management on the server side. But in the end, the user behind the password is still responsible for the password's strength. A poor choice can have dramatic consequences for the user or even for the service behind, especially considering critical infrastructure. On the other hand, law enforcement can benefit from a suspect's weak decisions to recover digital content stored in an encrypted format. Generic password cracking procedures can support law enforcement in this matter — however, these approaches quickly demonstrate their limitations. This article proves that more targeted approaches can be used in combination with traditional strategies to increase the likelihood of success when contextual information is available and can be exploited.

1. Introduction

Password-based authentication is older than modern digital society might realise. It is such an archaic system yet remains a crucial component of the security of most digital systems (albeit not necessarily the only one). Human beings are often predictable, and malicious actors are unfortunately able to exploit poorly chosen passwords to illegitimately enter into the system. While the consequences on a personal level can be severe, when a large service or critical infrastructure is targeted, the scale can become dramatic. It is therefore imperative to undergo a risk assessment when deploying a service requiring authentication and to make sure that all the precautions are taken.

1.1. Why are we still using passwords?

With all the known weaknesses of password-based authentication systems, one might wonder: why we are still using them? One explanation might be the public acceptance of this mechanism. Everybody has already used password-based authentication. In fact, it is considered that on average, a human has between 70 and 150 online password-protected accounts. But with that being said, what are the alternatives? Single-sign-on (SSO) strategies, active-directory, and password managers offer substitutes or enhancements over simple password authentication. Each of these solutions increases account security as they often require an additional element for a malicious actor to access a given system, e.g., having access to the key wallet protected by the password manager. Nevertheless, they still rely on a password at one stage or another and are unfortunately not yet widely adopted [1]. This is also largely true with two-factor authentication, where one of the factors often remains "something you know" – namely a password.

Lastly, some people might argue that they are no longer using passwords to unlock their phones, make payments, etc., but instead use a fingerprint or facial scan for identification. However, any service relying on a fingerprint reader or facial scanner on their phone can be bypassed by knowing the master code of the phone — this allows anyone to define a new fingerprint and bypass this security feature.

Therefore, passwords are not dead and will most likely continue to be used in one way or another for the foreseeable future. It is therefore of utmost importance to strengthen password-based authentication systems. Of course, this includes safe storage of the password on the server/device side. Furthermore, passwords selected by users should be strong in the very first place to ensure the best level of security.

1.2. Law enforcement investigation

The other side of the coin is that security reinforcements and the ready availability of strong encryption tools can also benefit criminal

[*] Corresponding author.
E-mail addresses: katerina.kanta@port.ac.uk (A. Kanta), iwen.coisel@europol.europa.eu (I. Coisel), mark.scanlon@ucd.ie (M. Scanlon).

https://doi.org/10.1016/j.jisa.2024.103809

enterprise and hamper lawful investigation [2]. Law enforcement agencies (LEAs) are nowadays encountering digital evidence in almost all investigations. An outstanding proportion of offenders, like any other member of society, have at least a mobile phone and a personal computer. These devices follow the security trends of the manufacturers and the content is most likely protected with a basic standard of protection at a minimum. Offenders often take additional security precautions if they are aware of the risks of investigation — as highlighted in the latest Internet Organised Crime Threat Assessment (IOCTA) report from Europol [3]. For example, they might employ additional levels of encryption over what might be enabled by default, such as full disk encryption or encrypted communication — again often protected by a password.

As stated by Plunkett et al. [4], "the lack of passwords, particularly during the execution of search warrants, has hindered investigations". It can be crucial to get access to such content during an investigation — necessitating the retrieval of the suspect's password(s). Of course, criminals are not always inclined to share their passwords with the investigators. It is not always possible to compel the suspect to divulge his/her passwords through a court order. For example, compelling password surrender could be considered as against the Fifth Amendment in the USA protecting suspects from self-incrimination [5]. In some other countries, it is considered a crime to not reveal a password under court order, e.g., in the United Kingdom within Section 49 of the Regulation of Investigatory Powers Act 2000. Nevertheless, the suspect may well decide to not reveal the password if the sentence incurred is lower than what might be expected should police gain access to the device(s). In each of these cases, LEAs have no other choice than to conduct password cracking processes to recover the suspect password and examine the targeted content [6].

The approach followed by digital investigators is diverse from those of malicious password crackers. The latter is predominantly interested in getting one hit to enter into a system under any user's account, or to gain access to the maximum of entries in a given dataset. The former are more interested in one specific user's account — the one of the targeted suspect. Therefore, it stands to reason that the cracking process can benefit from a tailored approach using the available contextual information of the suspect [7].

1.3. Contribution of this work

The work outlined as part of this paper proves that context plays a role during users' generation of passwords and can therefore be exploited by LEAs during their lawful criminal investigation. There is no dataset available focusing on a single user. As a result, the analysis outlined below is focused on a community level in order to extrapolate how likely a contextual-based approach is to succeed. Nevertheless, the bespoke, context-based approach outlined as part of this paper is proven to find passwords exclusively recoverable using this technique, i.e., those that were not found by currently used, generic approaches. The contribution of this work includes:

- An overview of the experimental methodology used in this paper, with a breakdown of the parameter definition process from dictionary creation parameters to password cracking tools selected.
- An extensive experimentation and results section, analysing the approach's performance across ten datasets of varying topics — proving the impact of context in password cracking.
- A thorough discussion of the uses, benefits and limitations of the contextual approach, as well as further optimisation steps in the future.

The remainder of the paper is organised as follows: Section 2 offers a brief literature review of related work in password cracking and dictionary creation. Section 3 presents the experimental methodology that was followed as part of this paper, with an analysis of the different parameters that were taken into account. Section 4 presents ten different experiments with targeted custom dictionaries compared to popular baseline dictionaries. Finally, a discussion of the results and possible avenues for future work are outlined in Section 6.

2. Background and related work

Human-chosen passwords have been analysed for more than a decade since the first data breaches including users' passwords have occurred. The most famous is the "Rockyou" data breach that happened in 2009. It is still today widely used in the literature for two main reasons. Firstly, it was at the time the largest data breach, exposing more than 32 million accounts with approximately 14 million unique passwords. Secondly, the passwords were not stored safely in the database, i.e., they were stored in plain text. This presented a significant advantage for password analysis, as the dataset also contains examples of the strongest passwords. More recent data breaches have exposed a significantly larger quantity of users. Plain text passwords are not typically exposed in more recent leaks and when encrypted passwords are leaked the strongest passwords in the leak are generally not retrieved.

2.1. Reinforcing password strength and management

Collected passwords from the aforementioned data breaches are useful for many fields of research, e.g., designing new password cracking processes, reinforcing password strength meters and models [8,9], etc. In all cases, data breaches represent a risk to the individual safety of each user concerned. Nowadays, passwords are almost always safely stored on the server side — whether they are stored encrypted or using a cryptographically secure hash function. Consequently, if the data leaks for whatever reason, undesired users would not immediately gain access to the password(s) of the user(s). The function used to safely store the password plays a crucial role at this stage, as it will directly impact the capacity of the malicious person conducting an offline attack.

For example, using a modern gaming graphics card, e.g., an Nvidia GeForce RTX 3090, in conjunction with `hashcat`[1] password cracking software, a malicious actor can try above 65×10^9 password candidates per second for a relatively weak hashing function, e.g., MD5. This number drops to around 100,000 per second in the case of a more secure password hashing function, e.g., Bcrypt (with 32 iterations). While the latter case offers more resistance to an attacker, a weak password in the Bcrypt scenario would still be found quicker than a strong one in the MD5 scenario. It is worth noting that the figures presented above are for a single graphics card and much faster performance can be achieved leveraging large GPU clusters, such as those offered by cloud computing providers, e.g., Amazon Web Services.[2]

Therefore, in terms of protection from password cracking and increasing security, passwords selected by users should be strong in the first place. Raising awareness is generally the best method to ensure that users understand the purpose and the importance of security and good password selection and management [7]. However, the reality is that users choose weak passwords and, even worse, often reuse their passwords [10–12]. This renders a strong password weaker in case it can be obtained from other sources by an attacker [13], e.g., keyloggers or phishing attacks. There are solutions to encourage users to not reuse the same passwords across different services, such as the "Have I Been Pwned"[3] service or the automated detection of password reuse in some Internet browsers.

[1] https://hashcat.net/.

[2] aws.amazon.com.

[3] https://haveibeenpwned.com/.

Password policies and/or password strength meters are often deployed to ensure a certain level of security. The most famous one is most probably the initial password policy designed by NIST in 2013 [14]. It requires passwords to be of a minimum length of 8 and to include lowercase, uppercase, special characters and digits. Such policies exist in many variations, but they tend to give a false impression of security. End users also tend to be predictable. For example, if the previous password was "password", they would simply capitalise the first letter and add a digit or special character at the end to make it compliant, namely transform it into "Password1!". Such a password is compliant to the password policy, yet it is not secure as password crackers have seen this pattern many times in data leaks of passwords and have adapted their attacks to mimic popular behaviour.

Password strength meters have therefore evolved to more advanced techniques, often analysing the inner structure of the password and detecting popular patterns. This is the case for example of the well-known `zxcvbn` tool [8] but also of many other academic solutions [9].

Users therefore need to create more complex passwords. Ideally, they should be long and random, but this makes them more difficult to remember. As highlighted in the analysis of passwords from 3.9 billion accounts [15], users tend to use password fragments that can potentially be linked to their context, such as a city or the name of a pet. Such facts provide a hook to design a context-based password-cracking attack if an individual is targeted, such as during a digital investigation.

2.2. Password cracking techniques

If users generated their passwords following a totally random distribution, no cracking strategy would have an advantage over an exhaustive, brute-force search — where all combinations of allowable characters of any accepted length are tested. This approach is guaranteed to work; the only unknown variable is how long it will take. Consider the aforementioned gaming card example testing 100,000 Bcrypt candidates per second, it would take approximately three months to try all candidates' passwords made up of digits, special, lowercase and uppercase characters up to length 6. This raises exponentially to 22 years for length 7, 2 millennia for length 8, etc. As a consequence, this approach is not preferred and will only be used as a last resort or if the investigator knows that the length is limited, allowing a full exploration in a reasonable time.

A time-memory trade-off approach relying on the principle of the Hellman table [16], Rainbow Tables [17], focused on mitigating the time required to explore a given space. For the price of some storage capacities and the pre-computation of the space to be explored, a password belonging to this given predefined space can be retrieved in a negligible time compared to the pre-computation step. This is highly valuable if one knows that several passwords are meant to be encountered in practice. Many projects have worked collaboratively to generate such rainbow tables, e.g., the rainbowcrack project[4] for the functions MD5, NTLM and SHA1. While efforts are still made to improve the performance for generating such rainbow tables, it is rendered almost useless in the field of password cracking due to the popular usage of a *salt* in the storage of passwords. A salt is a random string concatenated to the password before using it as the input to the hashing function. Theoretically, rainbow tables could still be built for salted passwords, but the defined space to explore must incorporate this salt. There is no fixed length for a salt, but it is generally long enough, e.g., 32 bits or more, to render rainbow tables no longer feasible in practice. Using salted passwords has the additional benefit that two identical passwords should have two different salts (as they are randomly generated) and will therefore have two different hashes in the database.

Fortunately or not, humans are often predictable. Human password selection is far from random and often follows the distribution of natural language [18]. Password trends are frequently analysed from leaked datasets and for years, the password appearing most often is "123456".[5] A dictionary of the most commonly used passwords can be designed from these predictable user tendencies. Typical trends, such as adding a '!' at the end or capitalising the first letter, can be easily combined with such lists of popular password candidates. These candidate modifications are referred to as "mangling rules" and can be generated automatically from data breaches, e.g., using the PACK suite of tools.[6] The input for these tools can be a list of passwords obtained from one or several data breaches or humanly designed on purpose.

More modern approaches rely on machine learning techniques analysing a given input set to produce a list of password candidates which can be combined with mangling rules, similar to dictionary-based approaches [19,20]. Those techniques include Markov-based models [21], probabilistic context-free grammars [22], and neural network based attacks [23]. One such example of a neural network are Generative Adversarial Networks (GANs); where a neural network is developed to create password candidates that fall as close to the distribution of real passwords stemming from real-world password leaks [24].

2.3. Password statistics

Whether considering dictionary-based approaches or machine learning ones, an input dictionary is required. This dictionary can be straightforwardly a single data breach or a concatenation of several data breaches. Amassing the largest amount of leaked datasets might seem like a good idea to detect password reuse. However, it is not optimal if one considers using mangling rules in combination with such input. Indeed, those breaches contain entries that, once modified, could be interpreted as junk. Maintaining a good balance between the size and the quality of the entries is a valuable activity that will increase the success rate and especially reduce the time before success. There are automated tools that help users to automatically sanitise wordlists, such as the *demeuk* tool[7] from the Netherlands Forensic Institute.

A more fine-grained analysis of the breaches can provide detailed information about users' tendencies and draft specific rules to mimic observed creation patterns. For example, when asked to create a password with lowercase and uppercase letters, users are most likely to capitalise the first letter of their password [15]. When asked to include numbers and/or special characters in their passwords, they are very likely to use number sequences such as '123', number repetitions such as '111', meaningful numbers such as '314', or use letter substitutions such as '@' for 'a' and '1' for 'i' [15]. Automated tools, such as PACK[8] or Pipal[9] can analyse datasets to produce a set of realistic mangling rules to be combined with a dictionary at a later stage.

General trends can be drawn from the analysis of large passwords datasets. Culture, education, social origin and religion have an impact during a user's password selection process. For example, AlSabah et al. [25] highlighted that Arabic users were three times more likely to include their mobile phone number in their password, while users from India and Pakistan were more prone to use names. According to another study, more than half of Chinese users use passwords only made of digits [26] and Chinese passwords have a different letter distribution, structure and semantic patterns compared to their English counterparts [27]. In a study of the linguistic and cultural impact on password cracking of three different language spheres, it was shown

[4] http://project-rainbowcrack.com/table.html.

[5] https://nordpass.com/most-common-passwords-list/.

[6] https://github.com/iphelix/pack.

[7] https://github.com/NetherlandsForensicInstitute/demeuk.

[8] https://github.com/iphelix/pack.

[9] https://digi.ninja/projects/pipal.php.

that when the attackers leveraged the knowledge of each language, the number of candidates needed to crack 10% of the passwords was significantly lower [28].

This contextual information, while being personal, can often be accessed using side sources of information [6,29]. This is particularly true in the law enforcement scenario; where the investigators often have some information about the suspect to hand. The target always remains human and as such, generic approaches should always be tried first to grab the low-hanging fruit. In a second stage, a more targeted approach could be considered. Yet, there is no automated manner to design a contextualised dictionary targeting a user or a community of users. Below, a method is outlined to generate contextual dictionaries, and it is demonstrated that it can be useful to crack passwords that could be missed by generic approaches.

3. Experimental methodology

In order to prove the role contextual information can play in password cracking, experimentation and analysis is needed. This section outlines the methodology used and each of the parameters used as part of the experimentation, and describes how/why they were chosen.

3.1. Baseline selection

Most dictionary attacks use wordlists that originate from one or more different data breaches. The passwords in these data breaches are sometimes leaked in their plaintext form, but more often they are hashed (and salted). This means that in order to make use of these lists in dictionary and/or other password cracking attacks, the passwords need to be cracked first. That is not always possible for 100% of the leaked data — meaning that it can be argued that some of these cracked lists do not contain the “harder to crack” passwords.

One list that does not fall in this category is called “RockYou” and contains 32 million passwords that were leaked in 2009. Because RockYou was leaked in plain text, it is a very popular wordlist that has been used by many researchers as a way to extract insights on users’ password habits [30,31] or as a baseline to compare other attacks against [32]. While the plain text passwords offer a significant advantage compared to an incomplete list from hashed leaks, one drawback of RockYou is that it was leaked in 2009. Since then, password policies around the globe have changed and stricter measures have been adopted — with passwords often needing to be longer and containing more than one upper/lowercase, number, and symbol characters. Therefore, in terms of adopting a baseline to compare the dictionary approach proposed as part of this paper against, RockYou was evaluated against a more modern dictionary, Ignis-10M.[10] Ignis-10M contains passwords from various leaks, and statistical analysis comparing its makeup against RockYou can be found on the project’s GitHub. Besides Ignis-10M, smaller versions of the Ignis wordlists were tested, but the 10M version achieved the best results. In terms of performance, RockYou and Ignis-10M had comparable results, with Ignis-10M performing slightly better. Therefore, Ignis-10M was chosen as the baseline for the rest of the experiments outlined below.

3.2. Dataset selection

Ideally, when talking about context-based decryption in a digital forensic setting, experimentation would be conducted on real cases by a digital investigator focusing on one specific target. But as this constitutes privileged/sensitive information, it is not possible to do this in a research context. Therefore, the focus is shifted in this paper to what is referred to as “the community approach”. In order to prove the importance of contextual information in password cracking using a community of users around a specific interest/topic, ten datasets were chosen from different online community leaks. These datasets come from `hashes.org` and their use for the purpose of this research has been approved y the Office of Research Ethics of [redacted for blind review]. As can be seen from Table 1, the datasets have been picked to represent a variable sample of topics and interests. These include data breaches from forums focused on music, cars, video games, recipes, and shopping. The datasets are all in English and contain only unique passwords, there are no repetitions. They are also of various length; the smallest being approximately 25,000 and the largest being 43 million — to encompass as big a variance as possible.

Table 1
The ten datasets involved in the experiments.

Dataset	Size	Date of breach
AxeMusic	252,752	N/A
JeepForum	239.347	November 2017
Minecraft	143,248	June 2015
MangaTraders	618,237	June 2014
Wattpad	23,531,304	June 2020
Battlefield	419,940	June 2011
Wanelo	2,130,060	December 2018
EverydayRecipes	25,271	N/A
Zynga	42,908,386	August 2019
DoSportsEasy	46,113	N/A

3.3. Password cracking methodology

In order to test the effect of context with these ten datasets, a pipelined approach to context based dictionary creation is adopted, as can be seen in Fig. 1.

According to this approach, a starting seed word is chosen that corresponds to a specific article on Wikipedia. For the purpose of this research, a structured version of Wikipedia, called DBPedia[11] is used. From the starting article in DBPedia, every link in the text of the article is visited and saved as a new entry in the dictionary list. Initially, the abstract and/or full article were also scanned and keywords were extracted, but for the most part these keywords coincided with the links within the article. For this reason, the decision was made to only consider the links, as the extraction of keywords from the article did not result in significant added value.

In the next step, these new DBPedia articles that are saved in the dictionary list, are then visited in turn and the same process is performed on them until a given depth from the original link is reached. For efficiency reasons, each of the explored links are stored in a set to avoid exploring the same input repeatedly. Subsequently, the links will be used to generate the dictionary. Some generic sub-strings inserted by Wikipedia are removed from the entry, e.g., “List of” or “Category:”. If the entry is composed of several words, the entry is saved as such but also a version without the stop words, if there are any contained therein.

These dictionary lists are then used as input with the password cracking tool of choice in order to crack the passwords of the data sets listed in Table 1. The password candidate creator tool that was chosen for this set of experiments was Prince,[12] as according to previous work by [32], it had the best performance. Prince is a password candidate generator that takes a dictionary list as input and outputs various combinations of the words in the dictionary list. This list of password candidates is then fed into John The Ripper[13] – the tool that performs the password cracking. To automate this process, the Password Guessing Framework[14] was used.

[10] https://github.com/ignis-sec/Pwdb-Public.

[11] https://www.dbpedia.org/.

[12] https://github.com/hashcat/princeprocessor.

[13] https://www.openwall.com/john/.

[14] https://github.com/RUB-SysSec/Password-Guessing-Framework.

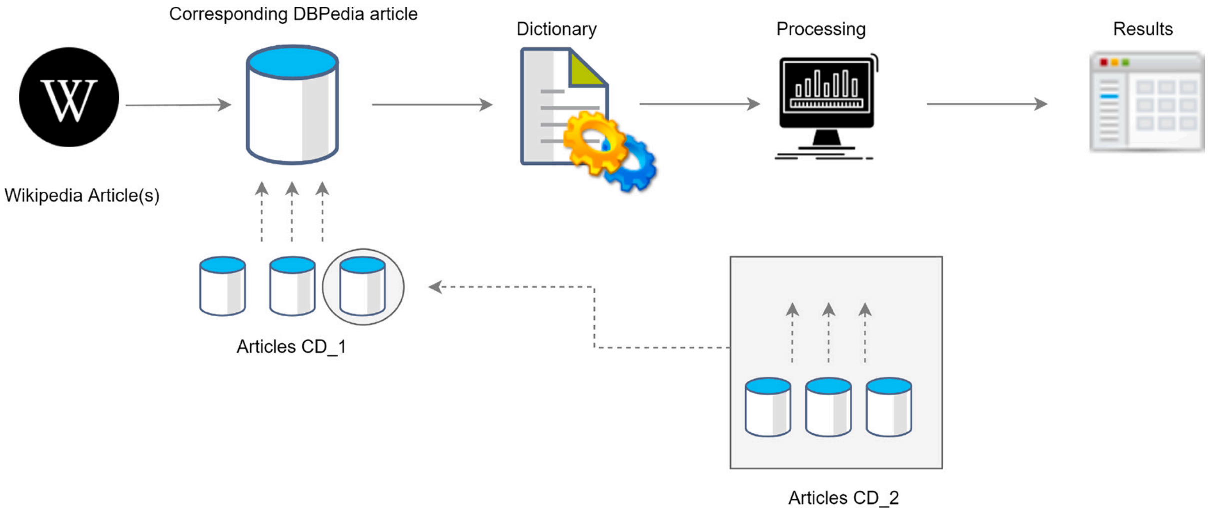


Fig. 1. Pipelined context based dictionary creation.

3.4. Parameter optimisation

For the purpose of this paper, ten seed words were chosen in order to correspond thematically to each leaked dataset, as shown in Table 1. These seed words can be found in Table 2. It should be noted here that the seed word for Battlefield is Battlefield_(video_game_series) in order to represent the video game Wikipedia article, but will be referred as simply Battlefield for the remainder of the paper. The seeds words were chosen to be as thematically close to the topic as possible. For example, for the Zynga leak, the world "Zynga" was also chosen as the starting point for creating the dictionary. For Wanelo, a leak from a website about shopping, the word "shopping" was used. As observed in the table, half of the seed words were chosen to be the same word as the target dataset, such as "Minecraft" and "Battlefield", while the other five were chosen to be a generic one-word description/category of the purpose of the website, such as "Cooking" for EverydayRecipes and "Sports" for DoSportsEasy. The rationale behind this was to study whether an identical seed word to the target leak over a generalised topic would be significant.

3.4.1. Generated dictionary level depth

The seed words mentioned above were subsequently used with the methodology outlined above in order to create custom dictionary lists. One parameter that needs to be defined at this stage, is the depth of these datasets, i.e., how many layers down from the seed word should be explored during dictionary creation. For this purpose, multiple dictionary lists for each seed word were generated; ranging from one layer to three layers, and in some cases four layers. Of course, the time to generate these dictionaries depends on the number of links in each level. The latency of the Internet connection has a significant impact on the speed to gather the pages from the online version of DBPedia. As indicative durations, Layer 1 is almost instantaneous, Layer 2 took ∼20 s, Layer 3 took ∼30 m and Layer 4 took approximately ∼1 day.

The performance of these varying layer depths was assessed for a selection of the aforementioned datasets and it was found that the dictionary lists produced by only 1 or 2 layers achieved lacklustre performance. For example, with the experimentation using the Wattpad leak, the custom 2 layer dictionary cracked 1.6% of the total passwords, while the 3 layer dictionary to cracked 42.1%.

A 4 layer dictionary was produced with the "Manga" seed word. This was used with the leak from Mangatraders, and it was still found the 3 layer dictionary performed better than the 4 layer one. More specifically, the 3 layer dictionary found 57.2% of the passwords, while the 4 layer one found 34.4%. This is due to smaller dictionaries facilitating more mangling for a fixed number of guesses than a larger one. Therefore, selecting a depth of 3 layers is the optimal choice. When keeping the number of guesses constant across the experimentation, it is important for the list to be long and detailed enough, but not too long as to include words that are too thematically distant from the seed word.

Table 2
The ten dictionaries produced by DBPedia.

Dataset	Seed word	Size
Axemusic	Music	1,001,173
Jeepforum	Car	853,825
Minecraft	Minecraft	243,803
Mangatraders	Manga	180,641
Wattpad	Fanfiction	641,007
Battlefield	Battlefield	415,311
Wanelo	Shopping	627,487
EverydayRecipes	Cooking	524,269
Zynga	Zynga	443,443
DoSportsEasy	Sports	31,918

Finally, as can be seen in Table 2, even though each of these datasets are of depth 3, their size varies according to how many links are contained in each Wikipedia/DBPedia page visited.

3.4.2. Password mangling rules

As mentioned in Section 2, password mangling rules are set during password cracking processes in order to imitate real users' password habits. For example, adding numbers or symbols at the end of a chosen password when the corresponding password policy requires them. These are generally useful and should be tailored according to the target. For the experiments outlined as part of this paper, the default mangling rules of John the Ripper were used on both the contextual dictionaries and the baseline dictionary.

3.4.3. Number of guessing attempts

When it comes to password cracking, the time taken to explore the password search space defined is directly related to the number of attempts permitted during the cracking phase's execution. Despite the brute-force cracking mantra of every password being crackable given enough time, this is realistically impractical in real-world scenarios. With a reduced search space and using a non-brute-force technique, more attempts will crack more passwords and/or have a higher likelihood of cracking a specific password — but at the expense of time and resources. As a result, password cracking typically requires a reasonable limit for the number of attempts to be decided upon.

In order to decide on the number of attempts to limit each experiment presented as part of this paper, a number of options were evaluated. To overcome the difference in dictionary sizes generated for a specific topic and/or generated dictionary level, a fixed size of guessing attempts was selected after experimentation and this was 10 billion. A lower number of guessing attempts produced worse results for both the baseline dictionary and the contextual dictionaries. On the other hand, more guessing attempts did result in more cracked passwords, but the trade-off between the additionally found passwords and the running time of the cracking process was deemed inefficient for the purposes of this paper.

4. Results

As mentioned in Section 3, ten datasets across various topics were chosen. Related contextual dictionaries with the help of Wikipedia/DBPedia were created. The depth of the dictionaries was selected as 3 and the number of guesses as 10 billion, as defined in the previous section. The cracking process over time for these ten datasets, with both the baseline dictionary (Ignis-10M) and the contextual dictionary, can be seen in Fig. 2.

From Fig. 2, it can be observed that for all ten datasets, Ignis-10M is the best performing dictionary. This is to be expected, as Ignis-10M is a compilation of different data leaks and contains some of the most popular passwords used by real-world users. Ignis-10M is also a 10 million entry dictionary, while the contextual dictionaries, as seen in Table 2 range from 1 million to 30 thousand candidates. It is therefore expected that Ignis-10M will perform better in comparison, and it will crack the most passwords across all different datasets as it is the most varied dictionary.

Focusing a little more into the varying results of the ten different contextual dictionaries (denoted as L_3s), it can be seen in Fig. 2 that Music_3 and Car_3 had some of the best performances, while Sports_3 had the worst. This can also be explained by the size of these dictionaries, with Music_3 being 1 million while Sports_3 is only 30 thousand. A dictionary of 30k candidates, even with the permutations allowed by 10 million guesses, cannot produce enough variance. This serves to highlight the importance of picking the correct seed word for generating a dictionary. The layer 3 dictionary Sport_3 was also generated, and it contained 1,068,758 candidates, which is a very significant increase over Sports_3. If “Sport” was used as a seed word instead of “Sports”, better results would be achieved when cracking DoSportsEasy, but the decision was made to use “Sports” as part of the experimentation outlined in this paper, to demonstrate the pitfalls of picking a bad seed word.

One interesting metric when it comes to the performance of these contextual dictionaries is how well they would do “stacked”, i.e., in a combination attack. To this end, Table 3 shows the percentage of unique passwords cracked by Ignis-10M and the L_3 contextual dictionaries. Column 3 of the table also presents the percentage of passwords that were only cracked with the L_3 dictionaries for each of the ten cases, i.e., the exclusively cracked passwords. Finally, Column 4 presents the improvement over Ignis-10M if it is combined with the contextual approach.

As can be seen in Table 3, in most cases the contextual dictionary has found approximately half the passwords found by Ignis-10M. Although in some cases, like JeepForum and EverydayRecipes, this number is even higher. Considering that Ignis-10M is compiled by a number of different data leaks and therefore contains actual used passwords across a range of services, the results of the L_3 dictionaries that are only dictionary words without any extra modification, is quite impressive. Once again, the only outlier is Sports_3, but this is somewhat expected since the input dictionary that was created from DBPedia contained only 30 thousand candidates. The passwords found exclusively by the contextual dictionaries offer on average an additional 2% of passwords, which in some cases represents a significant improvement over what was found by Ignis-10M alone.

Table 3
Total passwords cracked and improvement of the combination approach. The L_3 Excl. column contains passwords found only by L_3 dictionaries, while the L_3 Imp. column contains the improvement over Ignis-10M provided by the L_3 dictionaries.

Dataset	Ignis-10M	L_3	L_3 Excl.	L_3 Imp.
Axemusic	41.3%	20.5%	2.47%	5.97%
Jeepforum	68%	39.2%	2.32%	5.19%
Minecraft	38.4%	11.2%	0.76%	3.88%
Mangatraders	57.2%	28.2%	2.61%	4.56%
Wattpad	39.7%	15.2%	0.69%	17.86%
Battlefield	60.6%	29%	2.21%	3.64%
Wanelo	42.1%	19.3%	2.38%	5.64%
EverydayRecipes	64.4%	36.7%	2.24%	3.47%
Zynga	37.9%	15.7%	1.22%	10.61%
DoSportsEasy	41.7%	1%	0.06%	0.15%

Table 4
Class 3 passwords. The L_3 Excl. column contains passwords found only by L_3 dictionaries, while the L_3 Imp. column contains the improvement over Ignis-10M provided by the L_3 dictionaries.

Dataset	Ignis-10M	L_3	L_3 Excl.	L_3 Imp.
Axemusic	4504	581	286	6.3%
Jeepforum	3770	276	118	3.1%
Minecraft	3247	80	46	1.5%
Mangatraders	24,524	1906	942	3.8%
Wattpad	223,567	16,854	7758	3.5%
Battlefield	17,330	755	281	1.6%
Wanelo	47,604	3855	1709	3.6%
EverydayRecipes	254	41	24	9.4%
Zynga	417,404	33,752	15,735	3.8%
DoSportsEasy	934	6	1	0.1%

For example, with the Wattpad leak, while the passwords found exclusively by Fanfiction_3, represent a 0.69% increase, this translates to a 17.86% improvement over Ignis-10M. The reason for this is that while Ignis-10M finds more passwords, these are passwords that are repeated many times in the leak, while the passwords found by Fanfiction_3 do not have as many repetitions. This could indicate that the passwords found by Fanfiction_3 are less frequently chosen by users and therefore less encountered.

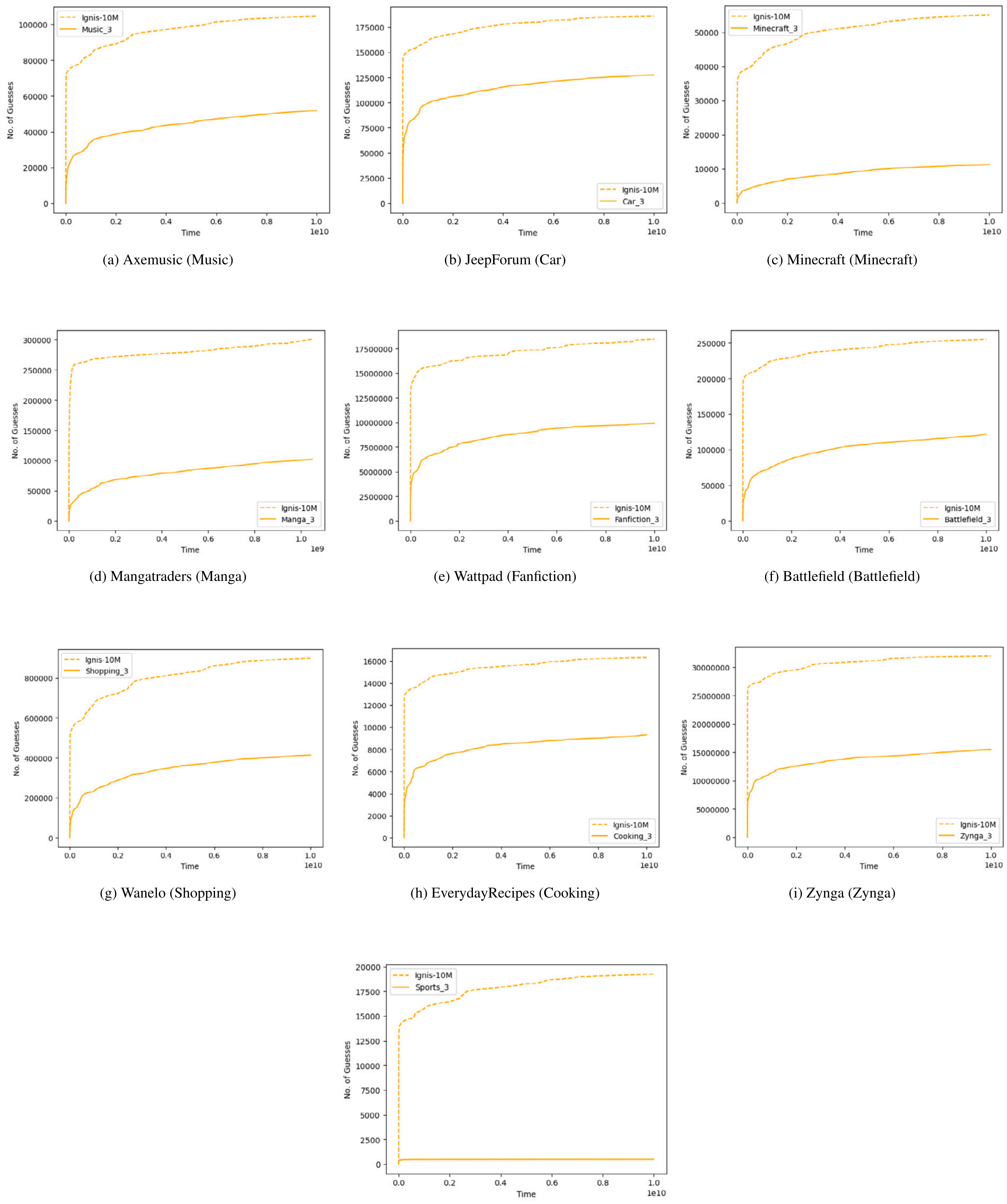
It can also be observed that the choice of either generalising the seed word or keeping it the same as the target dataset did not influence the results in any significant manner. Nonetheless, from the five best performing dictionaries, four were from the “generalised seed word” category.

Looking at the case of a single law enforcement officer wanting to gain access to an encrypted device, the number of popular passwords cracked from one data leak is not the optimal way to judge the effectiveness of a dictionary. In fact, if a suspect is hiding behind an encrypted device, it is reasonable that they are more tech-savvy, and it follows that there is a good chance that their password would be stronger than those found on the most popular password lists.

It is therefore important to also look at the quality of passwords cracked by the baseline dictionary and the contextual approach, i.e., the strength of these cracked passwords. To this end, Fig. 3 shows the breakdown of the found passwords by both approaches, classified into five classes of strength. The password strength meter used for this classification is `zxcvbn` and the five classes range from 0 to 4, with Class 0 being the weakest passwords and Class 4 containing the strongest.

As can be seen in Fig. 3, Class 1 and Class 2 passwords are those most commonly found, mostly from both Ignis-10M and the contextual dictionaries. This is because the passwords in these categories are easier to crack and would most likely be found by various approaches, as confirmed by [15]. It is therefore the Class 3 and Class 4 passwords that are the most interesting.

Tables 4 and 5 outline the passwords found by Ignis-10M, L_3, the exclusively retrieved by L_3, and the improvement percentage for Class 3 and Class 4 passwords cracked. By examining these two tables, it is



(a) Axemusic (Music)

(b) JeepForum (Car)

(c) Minecraft (Minecraft)

(d) Mangatraders (Manga)

(e) Wattpad (Fanfiction)

(f) Battlefield (Battlefield)

(g) Wanelo (Shopping)

(h) EverydayRecipes (Cooking)

(i) Zynga (Zynga)

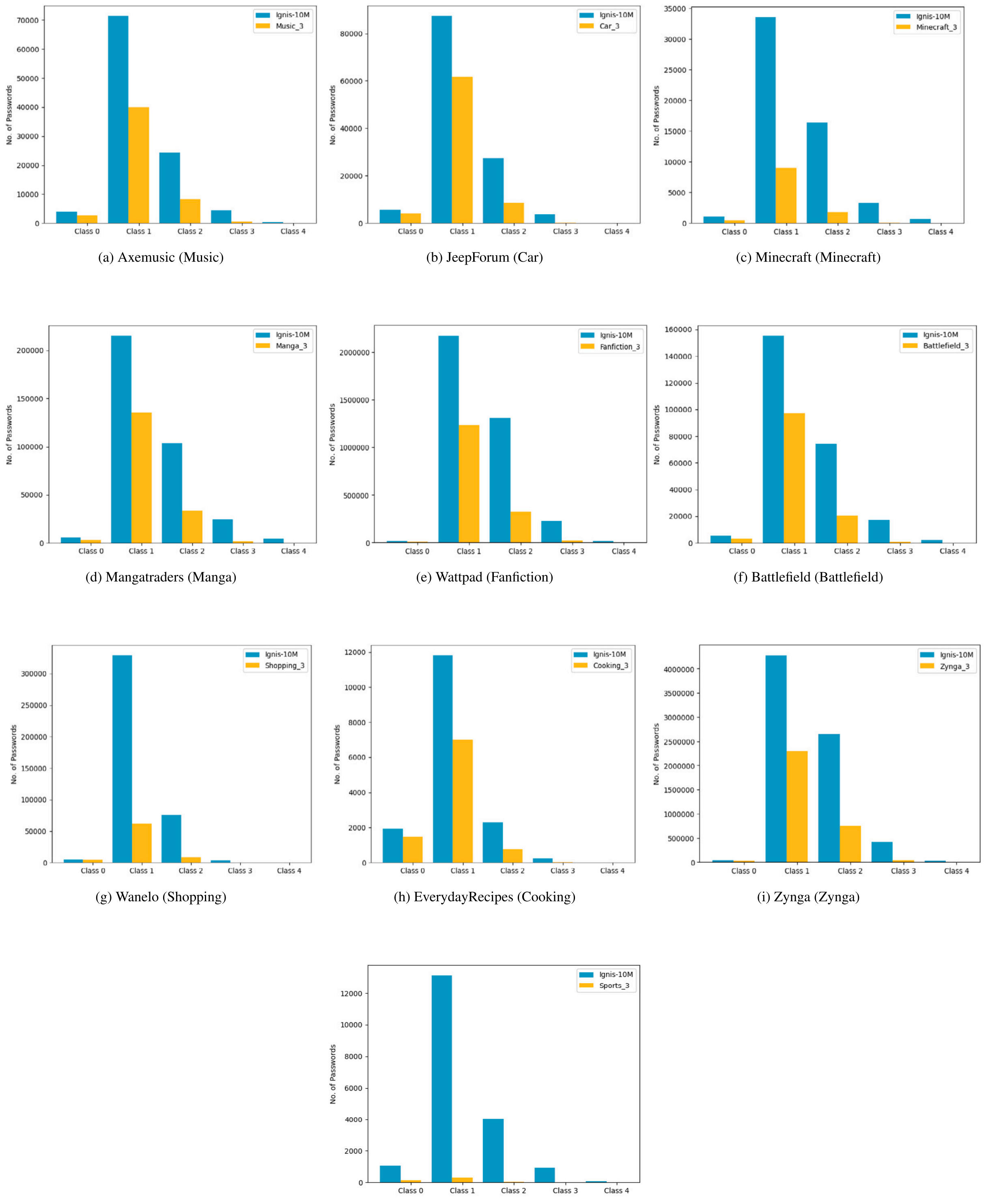
(j) DoSportsEasy (Sports)

Fig. 2. Passwords cracked by Ignis-10M and bespoke layer 3 dictionaries (Seed word for bespoke dictionary in parentheses).

notable that on average, approximately half the passwords found by L_3, are not found by Ignis-10M, cementing the importance of the proposed contextual dictionaries further. Furthermore, it can be observed that although the numbers are smaller compared to Class 3, Class 4 contains the strongest passwords and the percentage improvement of using L_3 on top of Ignis-10M is higher for Class 4. Notable examples are Fanfiction and Music achieving a 4.9% and 7.7% improvement respectively, In addition, for EverydayRecipes the percentage improvement is 42.8%, while acknowledging that the absolute numbers of recovered passwords are quite low for both.

5. Discussion

As the experiments of the previous section demonstrated, the added value of considering context in password cracking is evident. In a community-based approach, a bespoke, targeted dictionary can provide a significant increase in the number of found passwords and can be adopted ahead or alongside the baseline in the password cracking pipeline. Of course, one scenario that cannot be tested is that of a single suspect and their seized encrypted device(s). In such a case, a bespoke dictionary whose parameters can be tweaked and tailored to



(a) Axemusic (Music)

(b) JeepForum (Car)

(c) Minecraft (Minecraft)

(d) Mangatraders (Manga)

(e) Wattpad (Fanfiction)

(f) Battlefield (Battlefield)

(g) Wanelo (Shopping)

(h) EverydayRecipes (Cooking)

(i) Zynga (Zynga)

(j) DoSportsEasy (Sports)

Fig. 3. `zxcvbn` classification of passwords cracked by Ignis-10M and bespoke layer 3 dictionaries (Seed word for bespoke dictionary in parentheses).

the suspect can be created with ease using the proposed methodology and procedure as described in this paper. This would result in the investigator easily having the means to produce a custom dictionary, or dictionaries, for a specific case. When racing against the clock or when an encrypted device presents the largest roadblock in an ongoing case, contextual dictionaries tailored to the suspect at hand could prove invaluable to progressing an investigation.

6. Conclusion and future work

This paper offers an extensive experiment using ten different data sets of ten varying topics to provide a definite proof of the value of considering contextual information in password cracking. Humans are creatures of habit, and that is no different in their password selection process — where they often choose familiar words that are

Table 5
Class 4 passwords. The L_3 Excl. column contains passwords found only by L_3 dictionaries, while the L_3 Imp. column contains the improvement over Ignis-10M provided by the L_3 dictionaries.

Dataset	Ignis-10M	L_3	L_3 Excl.	L_3 Imp.
Axemusic	351	42	27	7.7%
Jeepforum	96	9	5	5.2%
Minecraft	667	5	3	0.4%
Mangatraders	4554	152	90	1.9%
Wattpad	15,022	1095	673	4.9%
Battlefield	2487	51	25	1.0%
Wanelo	2953	199	100	3.4%
EverydayRecipes	7	3	3	42.8%
Zynga	28,211	1403	849	3.0%
DoSportsEasy	60	0	0	0%

more easily remembered. This information can be leveraged in an investigation, and the ability to exploit it could prove invaluable during an investigation.

Of course, a contextual dictionary based around a single seed word cannot compete on equal footing with a 20 to 300 times larger and more well-rounded dictionary like Ignis-10M when the objective is to crack as many passwords as possible. This means that when no information is known about the target or the goal is to gain access to a system by cracking the password of any user and not a specific one, using a dictionary like Ignis-10M would provide a higher chance of success.

If the usage scenario surrounds a single case and/or a single password and information can be determined about its owner and their interested, then the contextual approach can be utilised. This would make even more sense, considering that in digital cases, suspects might be more likely to "try" harder to conceal their tracks and therefore would choose their password with more prudence.

The most notable improvement when it comes to the results of the contextual dictionaries is that for Class 3, and more importantly, Class 4, the number of extra passwords cracked with the L_3 dictionaries offers a significant improvement over the baseline. The extra passwords that were cracked not only lend credit to a combination approach, but also showcase further that a smaller dictionary built around one seed word related to the target data leak can indeed boost the number of cracked passwords significantly.

Something else to note, regarding the poor performance of Sports_3 compared to both Ignis-10M and the other L_3 dictionaries, is that when the size of the produced dictionary is too small, more layers or a different seed word (one with a more detailed Wikipedia page, hence more links) should be considered.

6.1. Future work

As part of future work, more consideration will be given to refining the generated dictionaries. Although 3 layer dictionaries provided the best results in 9 out of 10 experiments, this was not the case for the 10th. In a scenario with the bespoke dictionary is too short, additional seed words could be provided (or the wordlist expanded using a large language model [33]), and the resultant dictionaries could be merged, additional layers could be used, or a combination thereof. Moreover, these dictionaries are exactly that, dictionaries, whereas Ignis-10M contains real-world passwords. It is therefore important to look into ways of transforming the dictionaries into password candidates, with the help of well-refined mangling rules, that could better imitate the behaviour of users when they choose their password.

Another crucial step in that regard is to fine-tune the sanitisation process for the dictionary words. For example, a link could contain more than one word. Therefore, tweaking the manner in which these are combined could result in better password candidates. In this vein, trimming some branches during the dictionary generation process can reinvigorate the progress when relevancy declines. For example, during the exploration of layer 3, some candidates might be already too thematically distant from the seed word. If these candidates can be disregarded at this stage, they could give way to an exploration of deeper relevant layers rather than shallower irrelevant entries. As a result, the end dictionary may be the same length as layer 3, but would contain more relevant entries.

Finally, two other avenues to be explored for optimising the results of the candidate generation process are searching backwards during the link exploration process and ranking the dictionary entries. For example, if the seed word is "Manga", backward searching would include Layer -1, that is, all the pages on Wikipedia that link *to* Manga. This could provide a substantial addition of very relevant password candidates. As part of future work, a system for keeping track of how many times each entry was found could help indicate how relevant it is to the seed word and could be used to rank the resultant list.

CRediT authorship contribution statement

Aikaterini Kanta: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft. **Iwen Coisel:** Conceptualization, Methodology, Resources, Supervision, Validation, Writing – review & editing. **Mark Scanlon:** Methodology, Project administration, Resources, Supervision, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

References

[1] Kennison SM, Chan-Tin DE. Predicting the adoption of password managers: A tale of two samples. Technol Mind Behav 2021.

[2] Sayakkara A, Le-Khac N-A, Scanlon M. Electromagnetic side-channel attacks: Potential for progressing hindered digital forensic analysis. In: Companion Proceedings for the ISSTA/ECOOP 2018 Workshops. ISSTA '18, New York, NY, USA: Association for Computing Machinery; 2018, p. 138–43. http://dx.doi.org/10.1145/3236454.3236512.

[3] Europol. Internet Organised Crime Threat Assessment (IOCTA) 2023. Tech. rep., Publications Office of the European Union, Luxembourg; 2023.

[4] Plunkett J, Le-Khac N-A, Kechadi T. Digital forensic investigations in the cloud: A proposed approach for Irish law enforcement. Tech. rep., Science Foundation Ireland; 2015.

[5] Atwood JR. The encryption problem: Why the courts and technology are creating a mess for law enforcement. In: Saint Louis University Public Law Review. Vol. 34, 2015.

[6] Kanta A, Coisel I, Scanlon M. Harder, better, faster, stronger: Optimising the performance of context-based password cracking dictionaries. Forensic Sci Int: Digit Investig 2023;44:301507. http://dx.doi.org/10.1016/j.fsidi.2023.301507, URL: https://www.sciencedirect.com/science/article/pii/S2666281723000082. Selected papers of the Tenth Annual DFRWS EU Conference.

[7] Kanta A, Coisel I, Scanlon M. A survey exploring open source intelligence for smarter password cracking. Forensic Sci Int: Digit Investig 2020;35:301075. http://dx.doi.org/10.1016/j.fsidi.2020.301075.

[8] Wheeler DL. Zxcvbn: Low-budget password strength estimation. In: 25th USENIX Security Symposium. USENIX security 16, 2016, p. 157–73.

[9] Galbally J, Coisel I, Sanchez I. A new multimodal approach for password strength estimation—Part I: Theory and algorithms. IEEE Trans Inf Forensics Secur 2017;12(12):2829–44. http://dx.doi.org/10.1109/TIFS.2016.2636092.

[10] Bonneau J, Herley C, Oorschot PCv, Stajano F. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In: 2012 IEEE Symposium on Security and Privacy. 2012, p. 553–67. http://dx.doi.org/10.1109/SP.2012.44.

[11] Florencio D, Herley C. A large-scale study of web password habits. In: Proceedings of the 16th International Conference on World Wide Web. WWW '07, New York, NY, USA: Association for Computing Machinery; 2007, p. 657–66. http://dx.doi.org/10.1145/1242572.1242661.

[12] Stobert E, Biddle R. The password life cycle: User behaviour in managing passwords. In: 10th Symposium on Usable Privacy and Security. SOUPS 2014, Menlo Park, CA: USENIX Association; 2014, p. 243–55.

[13] Wash R, Rader E, Berman R, Wellmer Z. Understanding password choices: How frequently entered passwords are re-used across websites. In: Twelfth Symposium on Usable Privacy and Security. SOUPS 2016, Denver, CO: USENIX Association; 2016, p. 175–88.

[14] Burr WE, Dodson DF, Polk WT. NIST Special Publication 800-63 - Electronic Authentication Guideline. Tech. rep., National Institute for Standards and Technology; 2004.

[15] Kanta A, Coray S, Coisel I, Scanlon M. How viable is password cracking in digital forensic investigation? Analyzing the guessability of over 3.9 billion real-world accounts. Forensic Sci Int: Digit Investig 2021.

[16] Hellman M. A cryptanalytic time-memory trade-off. IEEE Trans Inform Theory 1980;26(4):401–6. http://dx.doi.org/10.1109/TIT.1980.1056220.

[17] Oechslin P. Making a faster cryptanalytic time-memory trade-off. In: Boneh D, editor. Advances in Cryptology - CRYPTO 2003. Berlin, Heidelberg: Springer Berlin Heidelberg; 2003, p. 617–30.

[18] Bonneau J, Shutova E. Linguistic properties of multi-word passphrases. In: Proceedings of the 16th International Conference on Financial Cryptography and Data Security. FC '12, Berlin, Heidelberg: Springer-Verlag; 2012, p. 1–12. http://dx.doi.org/10.1007/978-3-642-34638-5_1.

[19] Du X, Hargreaves C, Sheppard J, Anda F, Sayakkara A, Le-Khac N-A, Scanlon M. SoK: Exploring the state of the art and the future potential of artificial intelligence in digital forensic investigation. In: Proceedings of the 15th International Conference on Availability, Reliability and Security. Association of Computing Machinery; 2020, http://dx.doi.org/10.1145/3407023.3407068.

[20] Kanta A, Coisel I, Scanlon M. A novel dictionary generation methodology for contextual-based password cracking. IEEE Access 2022;10:59178–88. http://dx.doi.org/10.1109/ACCESS.2022.3179701.

[21] Narayanan A, Shmatikov V. Fast dictionary attacks on passwords using time-space tradeoff. In: CCS 2005 - Proceedings of the 12th ACM Conference on Computer and Communications Security. 2005, p. 364–72. http://dx.doi.org/10.1145/1102120.1102168.

[22] Weir M, Aggarwal S, Medeiros Bd, Glodek B. Password cracking using probabilistic context-free grammars. In: 2009 30th IEEE Symposium on Security and Privacy. 2009, p. 391–405. http://dx.doi.org/10.1109/SP.2009.8.

[23] Melicher W, Ur B, Segreti SM, Komanduri S, Bauer L, Christin N, Cranor LF. Fast, lean, and accurate: Modeling password guessability using neural networks. In: Proceedings of the 25th USENIX Security Symposium. SEC '16, USA: USENIX Association; 2016, p. 175–91.

[24] Hitaj B, Gasti P, Ateniese G, Perez-Cruz F. PassGAN: A deep learning approach for password guessing. In: Applied Cryptography and Network Security. Springer; 2019, p. 217–37.

[25] AlSabah M, Oligeri G, Riley R. Your culture is in your password: An analysis of a demographically-diverse password dataset. Comput Secur 2018;77:427–41. http://dx.doi.org/10.1016/j.cose.2018.03.014.

[26] Liu Z, Hong Y, Pi D. A large-scale study of web password habits of Chinese network users. J Softw 2014;9:293–7.

[27] Wang D, Wang P, He D, Tian Y. Birthday, name and bifacial-security: Understanding passwords of Chinese web users. In: 28th USENIX Security Symposium. USENIX Security 19, 2019, p. 1537–55.

[28] Mori K, Watanabe T, Zhou Y, Hasegawa AA, Akiyama M, Mori T. Comparative analysis of three language spheres: Are linguistic and cultural differences reflected in password selection habits? IEICE Trans Inf Syst 2020;103(7):1541–55.

[29] Kekü̈llüoğlu D, Magdy W, Vaniea K. From an authentication question to a public social event: characterizing birthday sharing on twitter. Proceedings of the International AAAI Conference on Web and Social Media 2022;16(1):488–99. http://dx.doi.org/10.1609/icwsm.v16i1.19309, URL: https://ojs.aaai.org/index.php/ICWSM/article/view/19309.

[30] Bonneau J. The science of guessing: Analyzing an anonymized corpus of 70 million passwords. In: 2012 IEEE Symposium on Security and Privacy. 2012, p. 538–52. http://dx.doi.org/10.1109/SP.2012.49.

[31] Kelley PG, Komanduri S, Mazurek ML, Shay R, Vidas T, Bauer L, Christin N, Cranor LF, Lopez J. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In: 2012 IEEE Symposium on Security and Privacy. 2012, p. 523–37. http://dx.doi.org/10.1109/SP.2012.38.

[32] Kanta A, Coisel I, Scanlon M. PCWQ: A framework for evaluating password cracking wordlist quality. In: The 12th EAI International Conference on Digital Forensics and Cyber Crime. ICDF2C '21, New York, NY, USA: Springer; 2021.

[33] Scanlon M, Breitinger F, Hargreaves C, Hilgert J-N, Sheppard J. ChatGPT for digital forensic investigation: The good, the bad, and the unknown. Forensic Sci Int: Digit Investig 2023;46:301609. http://dx.doi.org/10.1016/j.fsidi.2023.301609, URL: https://www.sciencedirect.com/science/article/pii/S266628172300121X.