Computers & Security 113 (2022) 102560

Contents lists available at ScienceDirect

Computers & Security

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

A Novel Password Policy Focusing on Altering User Password Selection Habits: A Statistical Analysis on Breached Data

Ebu Yusuf Güven[a,*], Ali Boyaci[b], Muhammed Ali Aydin[a]

[a] Department of Computer Engineering, Istanbul University Cerrahpasa, Istanbul, Turkey
[b] Department of Computer Engineering, Istanbul Ticaret University, Istanbul, Turkey

ARTICLE INFO

Article history:
Received 7 August 2021
Revised 4 November 2021
Accepted 21 November 2021
Available online 30 November 2021

Keywords:
Data Breach
Password Selection Habits
Password Patterns
Password Policy
Brute Force Attack

ABSTRACT

Online services generally employ password-based systems to enable users to access personal/private content. These services also force their users to change their passwords periodically under specific policies to increase security. However, analysis of breached data reveals that current policies do not consider user password selection habits and pose critical security and privacy concerns. Additionally, when passwords are leaked, attackers have the opportunity to study - and possibly identify - the structure or pattern of the user password selection set. This way, attackers could predict the next password or reduce the search space considerably in their attacks. Therefore, this study proposes a novel behavior-based password policy to increase the present security level and avoid further exploitations if a breach occurs. This study uses statistical methods and visualization techniques to examine the password selection behaviors of over ten million UserID-password pairs collected from anonymously shared data breaches. The data set is anonymized while keeping the uniqueness of userID-password pairs and shared with other researchers along with extracted features. Results show that user password selection patterns can be generalized and used to increase the success rate of attacks.

1. Introduction

Authenticating users with services is a critical step in ensuring data privacy and security. Different user authentication methods like token devices with varying interfaces of communication (such as NFS cards, USB Tokens), bio-metric information (such as fingerprint, cornea), and 2-factor authentication using two different methods sequentially provides different levels of security. However, text-based passwords continue to be the most common form Barkadehi et al. (2018). While all of these verification methods are perfectly acceptable, ease of use is a critical component of the user experience.

Text-based passwords in online services are typically used in conjunction with a userID that identifies a person (id, alias, telephone number, username, or email). Any organization or person providing a service over the web uses passwords to authenticate personal/private information access. Therefore, these service providers need some form of a password policy to make sure that users' passwords are sufficiently complex and hard to guess Wheeler (2016). However, attackers have developed a variety of countermeasures to weaken users' security, including brute force, dictionary-based, and rainbow table-based attacks Hu (2017).

Unfortunately, completing the authentication stage is not the only way to access confidential data. Attackers can gain access to user information by exploiting system security flaws Malderle et al. (2018). Additionally, users tend to utilize the same password for different services Choong et al. (2014). When one of the users' credentials (userID and password) is compromised, this information potentially leads to further exploitation of users. As a result, leaked or stolen data may be illegally sold and shared over anonymous file or text sharing platforms. These platforms may include cloud-based storage, code hosting services such as GitHub, and applications that enable anonymous sharing of files or text. This situation paves the way to establish an untraceable marketplace for trading or sharing illegally obtained user data. On the other hand, people can use these openly shared credentials to create services like "haveibeenpwd.com" and "breachalarm.com," which allow users to inquire about their userID/password leak status. These control systems with over 11 billion credentials create a working field for security researchers. However, this collection consists of only discovered or publicly available data. There is no doubt that the total number of leaked passwords is more exten-

* Corresponding author.
E-mail addresses: eyguven@istanbul.edu.tr (E.Y. Güven), aboyaci@ticaret.edu.tr (A. Boyaci), aydinali@istanbul.edu.tr (M.A. Aydin).

https://doi.org/10.1016/j.cose.2021.102560
0167-4048/

sive when undiscovered and attackers' privately kept data is considered.

On the other hand, password policies prevent users from selecting easy to guess or regularly used passwords. However, with enough password leaks, even these policies fall short of strengthening users' security. As it is not possible to prevent all security breaches, it is evident that more sophisticated techniques are required in password policies to secure users.

In the literature, it is observed that the focus is often on the password itself. Additionally, it is observed that the majority of password research is conducted via surveys, with the surveys generally remaining local. While research on password selection behavior with compromised data is quite interesting, the sources are usually not shared for privacy and security reasons. This study has produced the AuthInfo dataset that researchers can use in analysis and visualization and follow password selection trends.

Over ten million user records were gathered for this study from eight different anonymously shared sources on the internet. These resources are used to create AuthInfo dataset for analyzing user behaviors. The collected data is anonymized and made publicly available via GitHub platform for security researchers. In addition, user password selection behaviors are evaluated using statistical methods and visualization. The results demonstrate that statistical analysis of user behavior can help improve the success rate of brute force attacks against a user who has been a victim of multiple data breaches.

Outputs of this study are listed as:

- The analysis is done on userID and password pairs,
- Personal data is anonymized but preserved uniqueness within the dataset,
- Anonymized dataset shared as the AuthInfo dataset including newly created features from the analysis,
- Proposed a new security policy step to be used in selecting a new password using password selection behavior,
- Proposed a secure way to store password selection behavior of users,
- Demonstrated a step-by-step attack by learning password behaviors from previous data leaks.

This article is divided into the following sections: Section 2 examines password studies from various prevalent perspectives in the literature. The research methodology, including data collection techniques, pre-processing steps, anonymization method, and the AuthInfo dataset, is discussed in Section 3. Section 4 summarizes the analysis results. In Section 5, a method for service providers to strengthen their password policies is proposed. Following that, conclusions are given in Section 6. Finally, Section 7 explores future research.

2. Related Works

Analysis studies on user credentials (userID and password) and data breaches have been a field of study for researchers since the early days of digital technology. Generally, user credentials have leaks due to cyber-attacks caused by vulnerabilities of digital systems. These leaks give birth to an exciting new area for analyzing user behaviors. Data breaches are reported on various platforms, including breachalarm.com Yank (2021), which maintains a database of reported incidents, and haveibeenpwned.com Hunt (2021), enabling users to query their personal information from a pool of almost 11 billion records.

The integrity of leaked data, which is generally obtained illegally, is also questionable. Frequently, attackers modify or corrupt data intentionally to maintain an advantage Maschler et al. (2017). Also, attackers publish forged data to create distrust and harm individuals' or organizations' reputations. "Maschler et al." developed a software that verifies the authenticity of leak files and their associated records Maschler et al. (2017).

The majority of researchers have attempted to deduce user password selection habits through direct user surveys. For example, Jayakrishnan et al. stated that participants were forced to change their passwords regularly, which influenced their password creation strategies, and revealed that the new passwords are not more secure than the previous ones Jayakrishnan et al. (2020). While these studies enable users to analyze a variety of personal data, including occupation, education, gender, and nationality, their scope is limited due to the small sample size. Murphy's thesis examined the effect of password complexity requirements on the cryptographic strength of user-generated passwords. It was compared to a list of known passwords to determine the average Levenshtein distance Murphy (2018). Additionally, Murphy demonstrated in his thesis that a complex password policy encourages users to choose less secure passwords.

Creating effective password policies requires examining people's education, skills, and password management behaviors. NIST's report on password selection emphasizes security and usability concerns Choong et al. (2014). In the report, a questionnaire was distributed to gather data on users' password management practices for their work accounts. According to the report, similar password selection trends develop with password policies complexity. Also, other researchers who surveyed the behaviors and practices associated with password usage Awad et al. (2016); Shay et al. (2010); Wash et al. (2016) concluded with similar results.

Allowing users to choose weak passwords jeopardizes user security. Because attackers' initial attempt on online services without a known vulnerability in the login system is typically limited to brute force attacks Pagar and Pise (2017). These attacks target frequently used or default userID and password pairs such as "admin: password." Researchers have conducted studies to strengthen password security by providing feedback during the password setting process Shay et al. (2015); Wang et al. (2016). Despite strict security policies, passwords that have been frequently used are not controlled by all service providers and may also introduce security gaps.

Another employed method for password security is the honeyword Thite and Nighot (2021). Honeyword is an intrusion detection system used to detect attacks when a user defines a password combination that has been already known or can be generated about themself Belding (2018). A study was conducted to determine the honeywords, which revealed that Ergler attempted to use the service provider's dictionary list as a result of data leaks Erguler (2015). Thus, an intrusion detection study was conducted using frequently used and exposed passwords. As research on password trends and leaks expands, it is reasonable to expect that such honeyword works will expand as well. Additionally, such studies can use the AuthInfo data set generated from this study.

Password analysis of leaked user information reveals that region and culture influence users' password selection. Kvrestad et al. examined the frequency with which various character sets and keyboard patterns are used in passwords using leaked databases and a survey study Kävrestad et al. (2019). They concluded that the letter frequency of the American alphabet is still dominant in passwords. Studies are conducted on specific regions or languages when leaks involve local users and general password analysis. Maoneke et al. examined the effect of a multilingual password policy on user-generated passwords in African and Indo-European languages, inviting students from 224 African universities Maoneke et al. (2020). African languages are less successful with Probabilistic Context-Free Grammar (PCFG) attacks than Indo-European languages with short passwords. Wang et al. conducted an experimental study on these datasets by characterizing the passwords of Chinese users using PCFG- and Markov-Chain-

Table 1
Breach site and keywords for Google.

Breach Source	pastebin.com, anonfiles.com
Google Keywords	gmail, hotmail, email, password, VPN, account

based password attacks Wang et al. (2019). Even though Chinese web passwords are significantly weaker than their English counterparts, it has been demonstrated that the large character set makes attacks difficult Wang et al. (2019).

The leaked user information is analyzed and visualized using the features extracted from the userIDs and passwords. Previously published works concentrated primarily on password privacy and security Wu et al. (2019). Marthie et al. used extensive statistical data analysis to determine the effect of social identity and language on password and username in the Anti Public (AP) Combo list and "Exploit.in" (EX) datasets Grobler et al. (2020). While these studies generate valuable answers from large amounts of data, they do not provide generalized feature sets.

In this study, users' password selection habits are examined. In contrast to the literature, the analysis is done with the userID and password pairs. In addition, a novel behavior-based method is proposed to strengthen password policies. A step-by-step attack based on password behavior is also demonstrated using the proposed features. Additionally, the results and dataset are made publicly available for future research.

3. Methods

For attackers, leaked user data can be a significant source of income, and sharing data with no economic value is a method of spreading the crime. Attackers have been observed sharing leaked data via public services such as anonfiles[1] and pastabin[2]. When a user shares data over these services, the user is given a randomly generated URL. However, these services do not provide an index or an option to search the shared data, and only people who know this link have access to this data. Therefore, it is required to either wait for attackers to share these links or find a way to search through these platforms to find shared data breaches via these services.

In some cases, search engines such as Google index these services, and it is possible to access shared files with a simple keyword search. However, attackers also know and use the same method to conduct phishing attacks and malicious file-sharing against other users. Therefore, researchers should perform such a task in a secured sandbox environment.

3.1. Data Gathering

There is no standard method for collecting data breaches through free and anonymous data-sharing platforms. However, the data breaches shared by attackers can be accessed through search engines. It has been observed that breached data is frequently shared as text files with ".txt" extension. The search is done by combining the string ".txt" with relevant keywords, resulting in the discovery of multiple files. The used search terms for this study are given in Table 1. Eight distinct data sources were identified through Google searches for various sites and keywords such as "site:pastebin.com gmail". The resulting files are 178.txt, 99f.txt, b.txt, cf.txt, ei.txt, mc.txt, f.txt, and stc.txt.

The searching process is divided into three different stages: searching data breaches, getting raw data, and storing credentials as shown in Fig. 1. In the searching data breaches step, the results of search engines are obtained by the operator. Then, the results are saved to disk and sorted manually. This step is crucial as the located files can be in any shape and format. Also, false-positive results, including malicious ones, must be eliminated. This operation is done in a secure virtual environment. Selected files are processed by anonymization and feature extraction during the credential storing phase. Finally, the results are saved in a relational database.

3.2. Pre-Process Step

It is not straightforward to analyze and visualize data obtained from public sources, which are frequently stored in different formats. Although the format selection is limited (comma delimited, pipe-delimited, etc.), writing rules on the 10 million records significantly lengthens the operations. First, the data is cleaned and structured; later, frequently used features like length and character types are calculated in this step. All features used in this study are described in Table 3.

Also, the data may contain various encoding schemes and non-printable characters, and some records may be incomplete. These values are discarded from the dataset as outliers. Passwords with extremely short or long phrases are also considered outliers and removed from the data set to eliminate faulty records or non-human accounts.

The usernID and password fields must be identified in the file, as their location differs even in the file itself. Generally, sources have one column each for userID and password. However, some sources swapped the columns, and some sources had more than two columns. A script developed in Python programming language is used to identify the userID and password pairs. Then, these pairs are transferred to a relational database for analysis. Other rows are considered outliers and discarded. The workflow of this step is given in Fig. 2.

The password alone is not enough to analyze user behaviors. In addition to the password, it is possible to identify behavior in personal detail if the userID is included in the analysis. However, working with any user identifiable information involves legal obligations depending on the country and its regulations. For example, the Europen Union's General Data Protection Regulation (GDPR) suggests anonymizing any user-identifying personal data. Therefore, in this study, userID's which may identify a person directly or indirectly are hashed to create anonymity.

Hashing algorithms create a mapping from an arbitrary input to a fixed size output. This mapping creates a one-way transformation function. However, these algorithms are vulnerable to brute-force and rainbow table attacks and can be reversed given enough time. Furthermore, it is straightforward to generate hashes for usernames known to attackers. Therefore, to produce different outputs from known userIDs, a salt value (randomized string) is appended to userID to create a unique hash value for each user.

The MD5 algorithm is used to hash the userID, and a salt value is used as an additional safeguard to protect users' personal information. The anonymization operation is given in Eq. (1).

$$anonId = MD5((SubString(MD5(userID)0, 10) + Salt + userID) \quad (1)$$

Even if userID is hashed, this does not protect users personal information when they include their usernames in their passwords partially or fully. Twenty-nine users who used their userIDs as passwords and 483,869 users who used passwords containing a portion of their username have been identified during the analysis step. As a countermeasure, these users with such issues are excluded from the data set. These records add up to 4.83% of the

[1] anonfiles.com.
[2] pastabin.com.

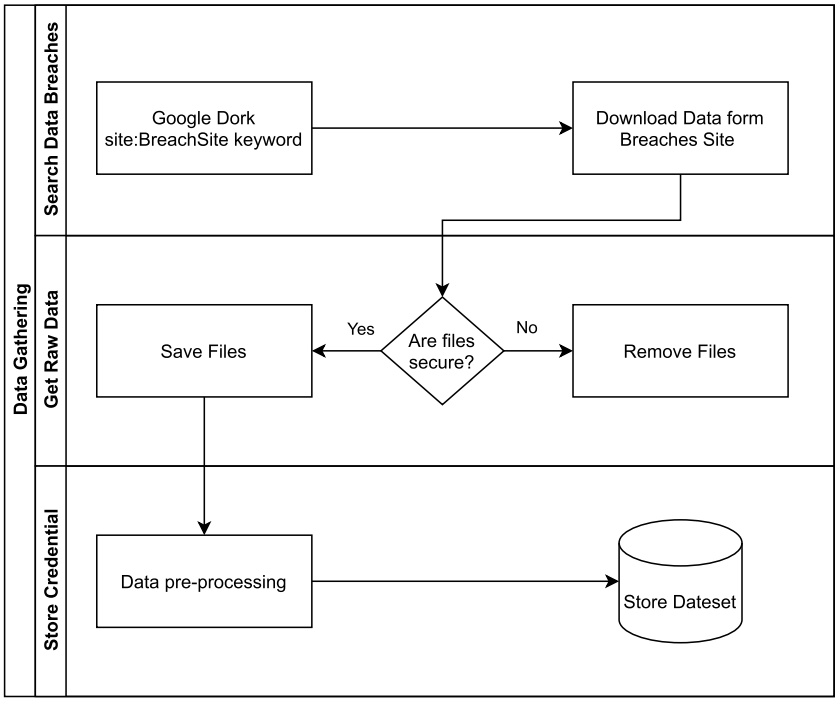

Fig. 1. Data Gathering Work Flow.

Table 2
Character types in the password.

Alias	Explanation	Values
d	Numbers	1,2,3,4,5,6,7,8,9,0
l	Lower Case English Alphabet	a, b, c, d, e, f,x, w, z
u	Upper Case English Alphabet	A, B, C, D, E,X, W, Z
s	Special Characters	%&/().:,

initially collected data. Also, any password shorter than four letters have been removed as they are too short for a secure password. A total of 5554 user records have been removed for this case.

Additionally, the most frequently seen userIDs are manually scanned during the pre-processing operation to ensure that only genuine user accounts are included. Any suspicious or generic userIDs such as “admin” are also removed from the data set to obtain consistent results.

3.3. Data Set

The AuthInfo data set is created by combining eight different anonymously shared data sources. These sources contain leaks of userID and password pairs from various services. After the pre-processing step, 10174482 userID and password pairs are used to compile the dataset.

While the data sources differ in the number of leaks and password policies, most of them demonstrated a similar distribution of password lengths. The password length distributions of the first four sources that account for most of the data set are given in Fig. 3.

During the pre-processing phase, frequently used proprieties such as password length and password mask are extracted from the data. The password mask shows the type of each character in hashcat application's[3] notations, which attackers widely use. Each character type is represented by a letter and given in Table 2 with their descriptions and scope.

[3] hashcat.net.

Additional features are derived from the username and password. Table 3 describes the extracted features, including the calculated password length, the character types in the password, and the mask created by replacing each character in the password with the appropriate character type.

The first and last character features have been added to help understand user behaviors and condense the password space for brute force attacks. This information can help to reduce the search space for brute-force attacks, especially with long passwords.

The AuthInfo dataset is shared on GitHub in CSV(comma-separated values) format for researchers to use GVEN (2021). Although the dataset is shared publicly, usernames are anonymized to prevent further user exposure. Furthermore, the userIDs are hashed with a randomized salt value to preserve the uniqueness of each userID. It is anticipated that researchers will use the dataset to confirm this study and to open up new avenues for future research. In addition, the data set is planned to be updated as new leakage sources are uncovered.

4. Results

The statistical extraction and data visualization studies performed on the data set demonstrate the characteristics and their relationship. Similar to other studies, “123456” appears to be the most commonly used password. Fig. 4 shows the most frequently used passwords in the corresponding data set. Also, “qiulaobai”, and “wmsxie” are regional passwords and widely used in China Wang et al. (2019); Yang et al. (2016). Regionality also gives hints about the location of the leak.

One of the most critical parameters in brute force attacks is the length of the password. Guessing the password length is crucial for a feasible attack since the password length has an exponential effect on required time (given in Eq. (2)). Fig. 5 illustrates the password length distribution for the dataset. The results indicate that passwords longer than 11 characters are relatively low, and users typically tend to select a password between six and eight characters in length. This behavior can be explained by the fact

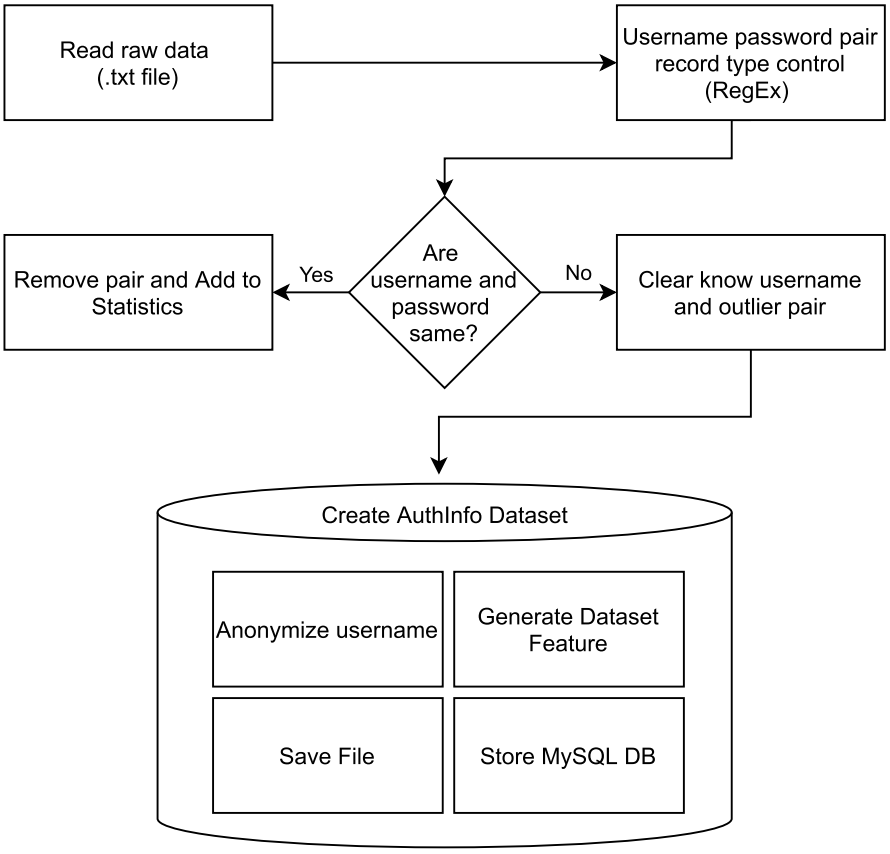

Fig. 2. Pre-processing work flow.

Table 3
Data Set Features.

Feature	Explanation	Sample
AnonId	Username hashed with salt	7CA2871... CD22FD40
Password	Password	Try34
Length	Number of password's character	6
Type of Char	Types of characters in the password	dlsu
Source	Data source alias	1. Data Source
Mask	The mask obtained by replacing each character in the password with the character type sign	ulldds
Type of First Char	Password start character type	u
Type of End Char	Password last character type	s
Number of Lower Case	Number of lowercase letters in the password	2
Number of Upper Case	Number of capital letters in the password	1
Number of Special Char	Number of special characters in the password	1
Number of Decimal	Number of digits in the password	2

that shorter passwords are easier to remember. Also, six is generally preferred as the minimum character length for many password policies.

The character types used in the password are another factor influencing password security in equal measure as the password length. A repeated permutation represents the total number of possible passwords composed of a given character set as:

$$P(n, r) = n^r \tag{2}$$

Where r is the password's length and n denotes the length of the character set chosen. Eq. (2) shows an exponential relation between the character set and the length of the password. For example, a password composed of numbers and 26 lowercase letters from the English alphabet with five characters will result in 60466176 possible passwords.

The distribution of passwords over the length and character types are given in Table 4. The data set suggests that most users who chose a six-character password did so with a number. However, users who prefer eight characters for their passwords favored a more significant proportion of numbers and lowercase letters. In this case, one could argue that service providers password policies requiring a password length of 8 compel users to choose a more

Table 4
Type of Password Characters.

Character Length	Decimal (D)	Decimal&Lower Case (DL)	Lower Case (L)	Other
6 Characters	1.418.316	384.005	207.169	23.597
7 Characters	972.478	714.900	216.970	43.950
8 Characters	732.477	1.208.874	198.165	51.933
9 Characters	538.461	928.638	214.342	29.254
10 Characters	284.824	573.458	99.160	35.653
11 Characters	257.041	398.385	50.727	20.447
12 Characters	43.212	168.040	40.559	15.386

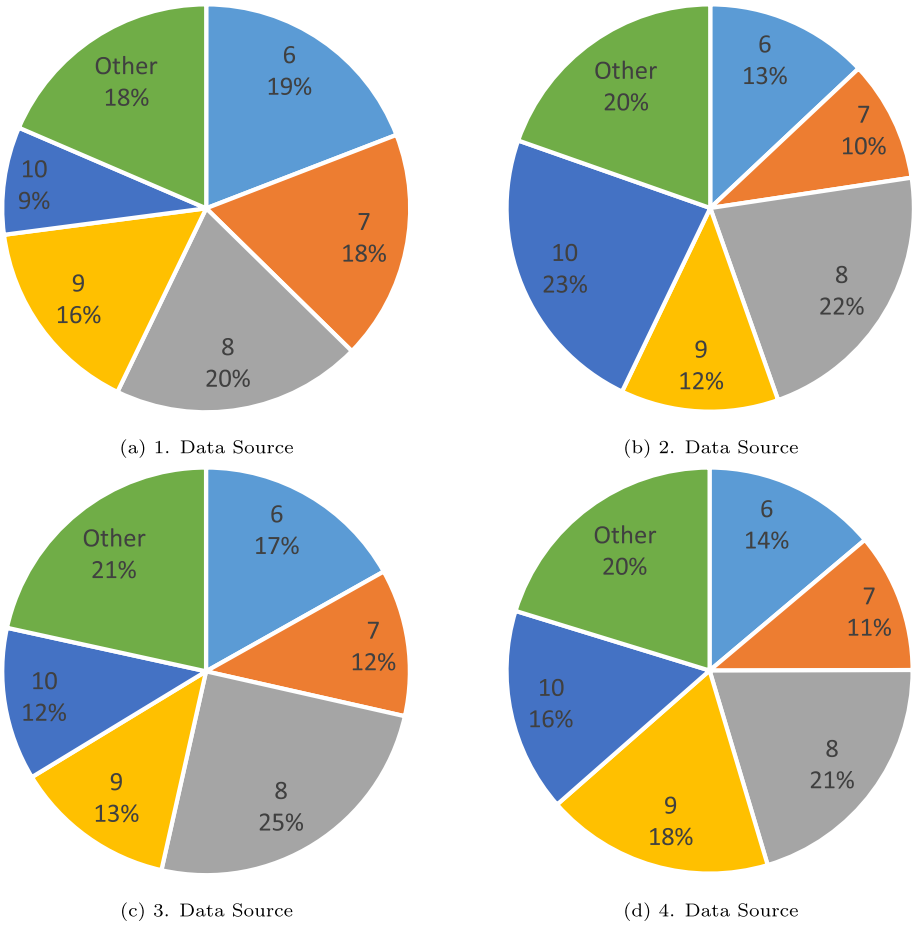

Fig. 3. The distribution of the first 4 source password lengths that make up the majority of the data set.

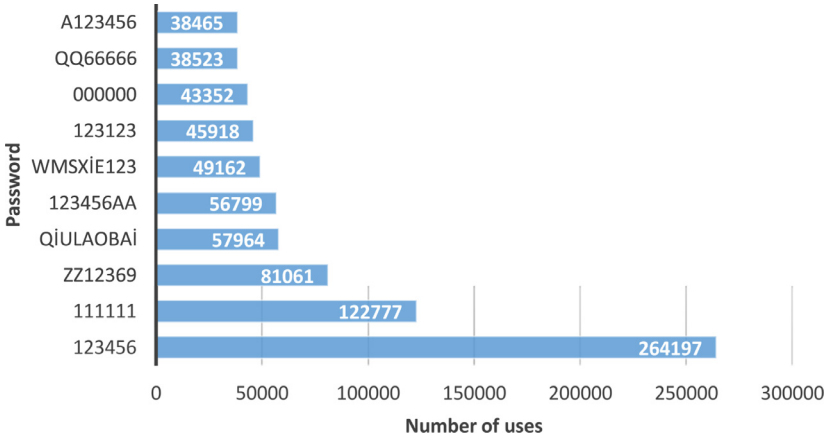

Fig. 4. Top 10 most used passwords in the data set.

secure password. Also, users who select a longer password have used a more comprehensive character set, indicating they are more aware of the importance of a secure password.

Values in Fig. 6 are normalized to the same percentage plane on a row basis to make user behaviors more understandable. As can be seen, the length of the password chosen by users is generally directly proportional to the complexity of the character type chosen. It is observed that as the length of the password increases, the probability of using more than two character types increases. Among the data set password combinations, it is observed that mixed character set (any character set combination except D, DL, and L) usage generally increases with the length of the password. The only exception is the passwords with 9 and 11 characters. This behavior can be explained by using numbers with patterns like birthdays, telephone numbers, or government ID numbers as passwords. After length seven, it is observed that more than half of users prefer the DL character combination.

When the dataset is analyzed, the most frequently used characters are listed in the following order: D, DL, L, UD, UDL, SDL, SL, SD, U, UL. Following D, DL, L, and UD, the remaining columns are combined into the “other” column to conserve space. The majority of passwords are composed entirely of decimal characters.

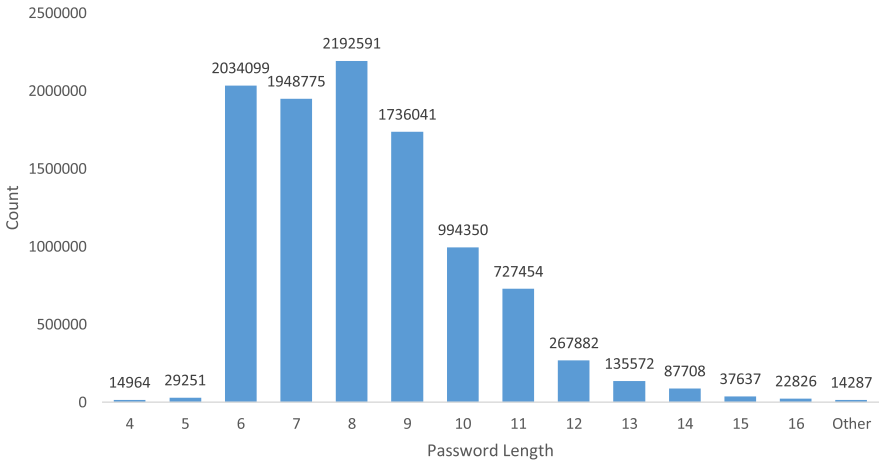

Fig. 5. Distribution of different password lengths.

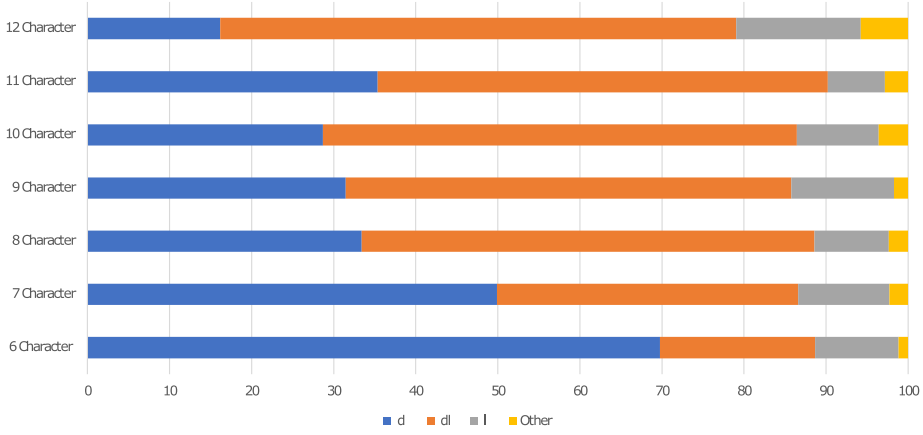

Fig. 6. Percentage distribution of character types within character lengths.

Table 5
Password Content Character Type Distribution.

Character Length	Lower Case	Upper Case	Special Chars	Decimal
6 Characters	%19,1392	%0,4542	%0,1187	%80,2879
7 Characters	%24,4381	%0,5713	%0,1315	%74,8592
8 Characters	%31,8647	%0,5179	%0,1719	%67,4455
9 Characters	%35,7715	%0,6651	%0,1713	%63,392
10 Characters	%38	%0,712	%0,2255	%61
All Password Lengths	%32,25	%0,67	%0,2	%66,88

This indicates that the sources' password policies are generally not stringent enough.

Password character type distribution is also calculated to discover which character sets are used for different password lengths. Table 5 shows the most widely used password lengths and their distribution over all sources. Also, the character type distributions for all password lengths are given in the last line of Table 5. It is clear that users want to utilize the minimum number of special characters as possible, and they often only use digits. Interestingly, when the password length increases, the users tend to combine special characters, upper and lower case letters, and numbers.

Identifying the possible first character of a password is advantageous since it narrows the search space of possible passwords for brute-force attacks. When some part of the password is known, the number of possible combinations can be shown as:

$$P(n, r) = d^m * n^{(r-m)} \quad (3)$$

Where n is the number of characters in the set, r denotes the password length, and d stands for the number of known characters. Thus, assuming password first m letters are known, the resulting possible set can be shown as Eq. (3).

Combining capital letters, lowercase letters, and decimals in the English alphabet (62 different characters), the information that the first letter of the password starts with a capital letter accelerates the attack 36 times according to the Eq. (3).

As most people tend to start a sentence with a capital letter, selecting a capital letter password is also expected to be common. First-character distribution percentages have been computed to understand how the users start their passwords. In Table 6, as the length increases, the use of lowercase letters increases, although the use of numbers decreases. The use of special characters and capital letters is observed less compared to other classes. For 11 characters length passwords, it is seen that the previously mentioned out-of-trend behavior continues in the first character selection. Therefore, for 11-digit passwords creating a separate evalu-

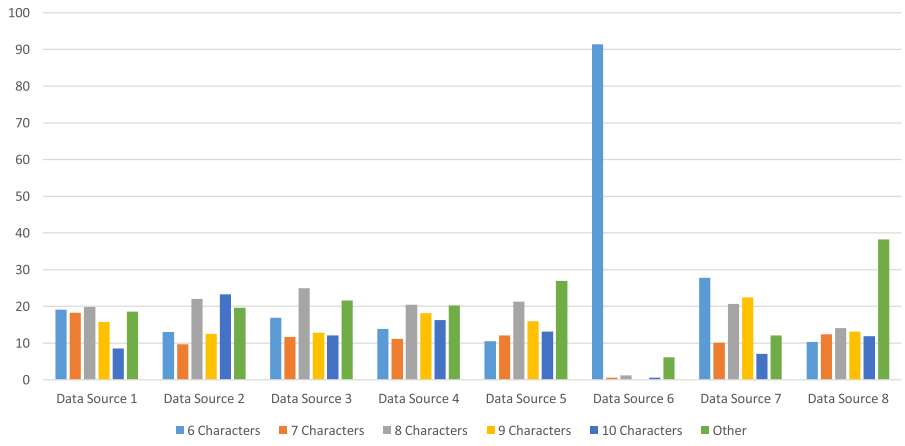

Fig. 7. Password length distribution percentages of data sources.

Table 6
Password First Character Distribution.

Character Length	Decimal (D)	Lower Case (L)	Special Chars (S)	Upper Case (U)
6 Characters	%72,98	%26,26	%0,053	%0,706
7 Characters	%55,15	%43,57	%0,029	%1,249
8 Characters	%46,48	%52,37	%0,046	%1,104
9 Characters	%40,99	%57,12	%0,040	%1,854
10 Characters	%38,95	%59,48	%0,045	%1,523
11 Characters	%60,03	%38,96	%0,027	%0,983
12 Characters	%24,11	%73,91	%0,060	%1,919
13 Characters	%9,66	%88,41	%0,053	%1,874
14 Characters	%9,49	%88,69	%0,068	%1,753
15 Characters	%11,97	%85,13	%0,061	%2,834

Table 7
Password first character distribution.

Mask	Count	Length
dddddd	1418316	6
ddddddd	972478	7
dddddddd	732477	8
ddddddddd	538461	9
ddddddddddd	398385	11
lldddddd	324169	8
ddddddddddd	284825	10
lllllll	216930	7
lllllllll	214308	9
llllll	207165	6

ation will increase the effectiveness of the attack. For users who prefer passwords with 12 characters or more characters, it has been observed that the use of digits in the first character drops below 10%.

Nevertheless, the use of digits in the first character trend increases again after 15 characters. It is seen that the tendency to use lowercase letters in the first character of the password generally increases up to 15 characters and decreases to 38% in passwords with only 11 characters. Additionally, over 90% of users who use capital letters in their passwords prefer the first character. When the last character of the password was evaluated, 30% of users who used special characters used it as the last character.

The distribution of the number of characters in the passwords is another critical parameter in password security. A commonly used character set and length information can significantly reduce the brute force attack trial space. Table 7 contains a distribution of password character type masks that can be evaluated independently.

Fig. 7 shows the length distributions of the sources included in the data set. While the distributions of data sources are generally balanced, the distribution of six-character lengths in data source six is highly concentrated. This anomaly can be interpreted as the password policy for the corresponding resource.

5. A Novel Password Selection Approach

Password policies are developed to ensure the users select a secure enough password. Various metrics have been developed for these password policies, such as character length and the use of different character sets or distance of keys on the keyboard Wheeler (2016). Additionally, some policies save old passwords (presumably hashed versions) and force users to select a different password from their previous ones. Also, as an additional measure, some internet service providers (such as Github and Google) check for password leaks for user accounts in addition to their current security policies. However, it is shown in the case study part of this work that attackers can learn password selection behaviors if passwords are exposed multiple times. Even if the user's password is different from their previous ones, combining the user behavior with a brute force attack significantly shortens the required time. Therefore, it is recommended to change the behavior along with the password. As far as the authors' knowledge, such a study has not been done before.

In addition to the "Compliant with Password Policy," "Different from the Previous Three Passwords," and "Different from Related Password Leaks" steps shown in Fig. 8, a "Different from the Previous Password Behavior" step is proposed to create a comprehensive password policy.

Two steps are required to enable the "Different from previous Password Behavior" feature. The first step is to detect the behavior, and the second is to store it securely.

Password selection behavior must be evaluated during password creation time, as the password will not be available to analyze later. However, keeping the password behavior raw in the database will also create a security risk in case of a possible leak. Therefore, the password behavior should be hashed with other information to ensure users' confidentiality and track their previous behaviors. It is also possible to provide the uniqueness of the output among different services by adding a salt value to these features as shown in Eq. (4).

$$PasswordBehavior = hash(userID + Mask + \epsilon \times Salt) \quad (4)$$

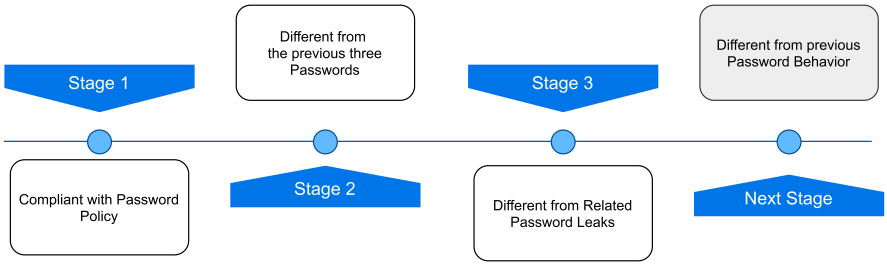

Fig. 8. Secure password selection control steps.

One way of generating password behavior information is using mask data by replacing the characters with character types in the password. However, because the mask information alone can reveal user passwords, this data should be combined with additional user information. As an alternative, a hashed version of the userID used in this study, AnonID, can achieve similar behavior. Finally, an optional unique salt value ensures the diversity of the same user data on different services.

The "Different from previous Password Behavior" step does not require additional information beyond what is already used in other stages. With minor modifications, existing password policies can quickly adapt to the proposed feature. Additionally, a technique like in Eq. (4) increases the security of this information even leaks happen.

6. Conclusion

This study uses statistical and visual analysis of leaked user credentials to understand user password selection behaviors and then proposes a novel password policy based on this information. Over 10 million userID-password pairs are collected, anonymized, examined, and processed to create the AuthInfo dataset. Each step of this process is defined in detail and supplied with flow charts to establish a method to share personal data while protecting user privacy and security.

AuthInfo analytics demonstrate that it is possible to infer the password policies of leaked data sources. Additionally, the most commonly used passwords, such as "123456," indicate that current password policies are ineffective. Further, it is demonstrated that nearly half of the passwords collected in AuthInfo are between six and eight characters in length and that the minimum requirements in password policies greatly influence user behaviors. Finally, it is observed that there is a direct relation between password length and usage of multiple character sets, which results in more complex passwords.

Additionally, users exhibit a preference for certain character types, positions, and frequency of use, similar to that of commonly used passwords. Furthermore, the users in the AuthInfo dataset used mostly numbers and lowercase letters with different lengths and rarely utilized capital letters and special characters. Also, over 90% of users who use capital letters utilize the first character as a capital letter in their passwords. Similarly, when a special character is included in the password, 30% of passwords use it as the final character. As demonstrated in Appendix 1. Case Study, these statistics can assist attackers in launching more successful brute-force attacks. These findings indicate that even when password policies are sufficiently strict, the security of users who exhibit similar password selection behaviors is reduced.

On average, a user has six different accounts across multiple internet services. Even if the users choose a different password for all these services, attackers can learn their behaviors individually with enough password leaks. The Authinfo dataset contains users with multiple leaks to illustrate this situation. A randomly selected user with seven different password leaks is used as a case study to show an incremental attack. It is observed that users generally stick with a pattern to create their passwords. Assuming the password selection pattern will be similar, an attacking strategy is devised. It has been demonstrated that an attacker may finish a seemingly unfeasible brute force attack in mere hours. This shows attackers can guess users next password relatively easily with enough data.

Password policies force users to choose secure passwords for their security. Many such policies require different character types and a relatively long password. Additionally, some services check if the user employs an already compromised or a common password. However, even a relatively good password policy does not help users when a data breach occurs if they do not require users to change their password selection patterns with each new password. Therefore, a new method is proposed to store and use password patterns by adding a new phase to password policies. The proposed method can easily be integrated with existing software with minimal modifications to strengthen policies further.

Finally, the AuthInfo dataset, consisting of userID, password, and additional features, is created and shared publicly on Github GVEN (2021) for the use of researchers. To protect users' confidentiality and privacy, personally identifiable elements are anonymized. Additionally, the repository will be updated and expanded as additional data leaks are discovered.

7. Future Work

One of the challenges in Password selection behavior studies is the availability of datasets. AuthInfo is expected to cover the base needs for these studies. However, with technology, the security requirements along with attackers' strategies are evolving rapidly. Therefore, the AuthInfo dataset will be updated frequently to catch up with these needs and increase the number of features on the data set. In addition, researchers are encouraged to contribute to the Github repository because a larger dataset enables the detection of more patterns like commonly used prefixes and suffixes and discover complex patterns.

Multiple leaks of the same user can reveal their habits. The minimum required amount of leaks for a user to be exposed can help on risk analysis for personal or organizational levels. It is possible to develop a probabilistic method to evaluate the relationship between password leaks and the risk of user behavior exposure.

Password policies should be improved by additional steps such as pattern and entropy analysis of passwords to make them more robust and resilient to leaks. Additionally, these improved password policies can be offered as a cloud-based service for websites or local applications for individual users.

Finally, end-users should be warned and educated about the importance of using more complex passwords to protect their privacy and security. No amount of policies can protect a user if they

do not care. Therefore, curriculums should include fundamental security and privacy techniques to help educating the public.

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.

CRediT authorship contribution statement

Ebu Yusuf Güven: Conceptualization, Methodology, Visualization, Software, Data curation, Writing – original draft. **Ali Boyaci:** Visualization, Methodology, Investigation, Validation, Writing – review & editing. **Muhammed Ali Aydin:** Supervision, Validation.

Acknowledgment

This study is prepared using the infrastructure of the IoT Security Test and Evaluation Center (Project Number YMP-0061), supported by the Istanbul Development Agency, within the scope of the thesis titled “Development of a New Scan Model for Cyber Threat Intelligence” in Computer Engineering Department at Istanbul University-Cerrahpaa Institute of Graduate Studies.

Appendix A. Case Study

Leaked passwords provide insight into password selection and policy in general; a tailored approach can lead to attacks for individual users. Individuals with multiple leaks are especially susceptible as the attackers can easily extract their habits. According to the NIST report, people use an average of six different online services, which significantly increases the likelihood of multiple passwords being leaked Choong et al. (2014). AuthInfo shows that the percentage of users who have encountered multiple leaks is close to 1 in 1000. Figs. A.9 and A.10 show various password leaks in the data set for the same user. The vast majority of the multiple leaked users have two or three leaks. With this information, a step-by-step brute force experiment shows that it is possible to learn the password selection behavior for a selected UserID.

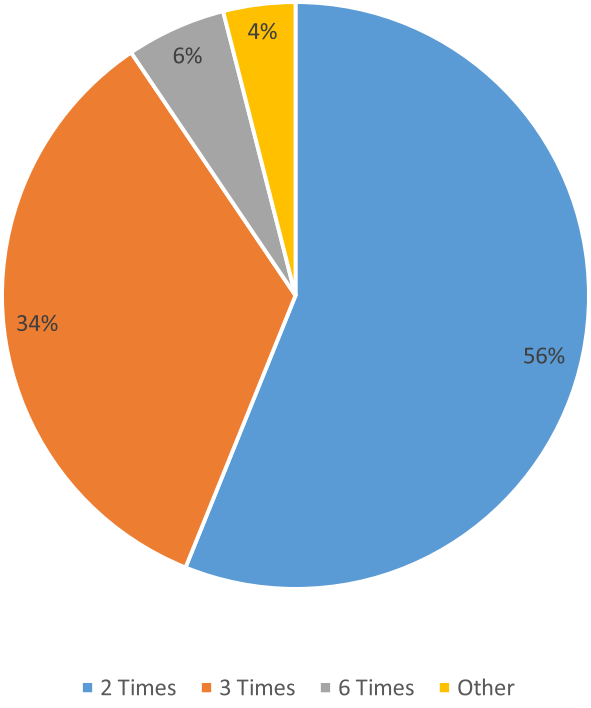

Fig. A.9. Distribution of multiple leaks.

This experiment assumes that the leaked passwords are hashed with the MD5 algorithm without salt values. The strategy of the attack is as follows: initially, the brute force attack is conducted blindly (without prior information). Then, after each successful attack, the resolved password is added to the known passwords database for the user. Also, the solved passwords are used to repeat the attack with more information about the user.

The MD5 challenge uses to evaluate the brute force attack’s performance. The computing environment is a High-Performance Computer (HPC) environment with 250 GB RAM, a 2.40GHz Intel (R) Xeon (R) Gold 6148 CPU, and 4 Tesla T4 GPU hardware with 14528/15205 MB and 40MCU. Additionally, Hashcat application is used for cracking the MD5 strings.

An individual user whose password has been leaked ten times is selected at random, and the users passwords are given in Table A1. The passwords are hashed with the MD5 algorithm without a salt value. Also, the attacker did not know any prior information like known previous passwords about the victim. The first parameter for a brute force attack generally is the password length. The limit of Hashcat application is nine characters when all printable ASCII values are used. Therefore, the size is set to nine characters. Next, the attacker tries to guess the hash of the relevant user’s password without utilizing prior information. Hashcat calculates 287 days and 13 hours to complete a brute force attack using lower and uppercase numbers and special characters. Due to time constraints, Hashcat was terminated after around one week. The attacker tries all character sets between 4 and 8 characters and fails during a brute force attack. It has been observed that the password of the victim user has a length of at least nine characters. The executed command is given below.

```
Challenge: ab0ccdfc2bf732ec09b94ec8221e0c39 and Other 10 MD5
    Hash
hashcat -a 3 -m 0 -i --increment-min=4 hash.ebu ?a?a?a?a?a?a?a
    ?a?a
output: none, Operator Canceled
Estimate time: 287 days, 13 hours
```

The next attempt is made with the Authinfo dataset to narrow the attack space using the results of this study. The attacker sees that close to %80 of AuthInfo dataset leaked passwords are between 4 and 10 characters long. Additionally, more than %85 of these passwords are numeric and lowercase character types. Using this information, the attacker can crack the eight-character passwords easily. For this operation, the command and the output are given below.

```
Challenge: ab0ccdfc2bf732ec09b94ec8221e0c39 and Other 10 MD5
    Hash
hashcat -a 3 -m 0 -i --increment-min=4 -1 ?d?l hash.ebu
    ?1?1?1?1?1?1?1?1?1?1
Output: cambridg1
Estimate time: 7 mins, 37 secs
```

After exhausting ten characters for lowercase and numeric character sets, the attacker determines that other password lengths are longer than ten. When the attacker reviewed the AuthInfo analysis for 11 characters, the attacker discovered that users’ password choosing habits included character sets of 52% decimal and lowercase, 36% decimal, and 9% lowercase. The attacker initially attempted a brute force attack on the decimal or lowercase charac-

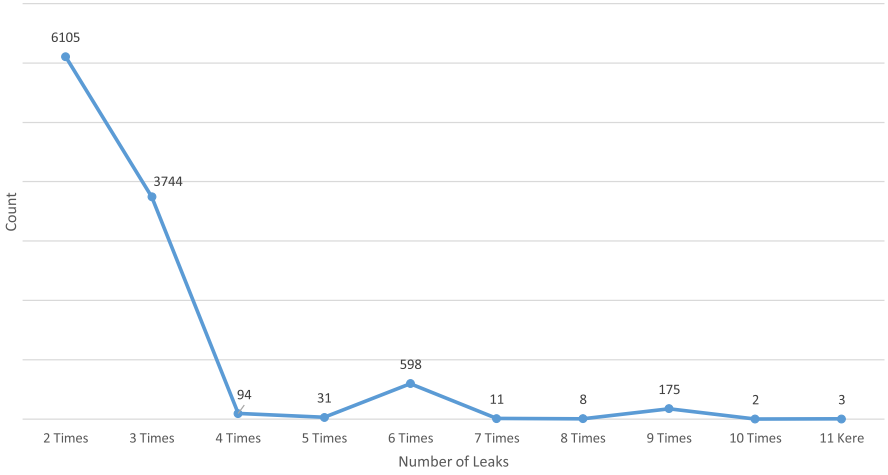

Fig. A.10. Distribution of users exposed to multiple leaks.

Table A1
Password choices of a user with the username dce9fef8b5c3ba9d576ee9ad37259632.

Password	Lenght	Type of Char	First Char	Count L	Count U	Count D	Count S	Mask
cambridg1	9	dl	l	8	0	1	0	llllllld
cambridgedw	11	l	l	11	0	0	0	lllllllllll
cambridged1	11	dl	l	10	0	1	0	llllllllld
cambridged0w	12	dl	l	11	0	1	0	lllllllllldl
cambridged0wq	13	dl	l	12	0	1	0	llllllllldlll
cambridgeasdaq	14	l	l	14	0	0	0	llllllllllllll
cambridge12177	14	dl	l	9	0	5	0	lllllllldddddd
cambridge123qqq	15	dl	l	12	0	3	0	lllllllldddlll
cambridgeazasd1d	16	dl	l	15	0	1	0	lllllllllllldl
cambridge212311q321qwe	22	dl	l	13	0	9	0	lllllllldddddddlddddlll

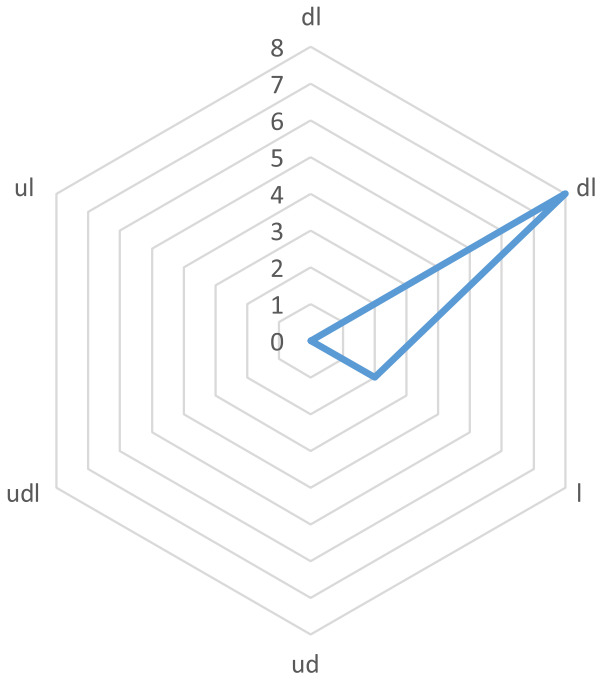

Fig. A.11. Pattern distribution of victim users nine solved passwords.

ter sets for 11 characters, which took approximately two months for the decimal and lowercase character sets. Finally, the attacker failed for the 11-character length decimal character set. The attacker succeeded in breaking the hash of “cambridgedw” for 11 characters using a lowercase characters set brute force attack. For this operation, the command and the output are given below.

```
Challenge: 4137b9bce6c0e826abf062cb677a1498 and Other 9 MD5
    Hash
hashcat -m 0 -a 3 -1 ?d?l hash.ebu ?1?1?1?1?1?1?1?1?1?1
Output: None, Operator Canceled
Estimate time: 51 days, 2 hours
Challenge: 4137b9bce6c0e826abf062cb677a1498 and Other 9 MD5
    Hash
hashcat -m 0 -a 3 hash.ebu ?d?d?d?d?d?d?d?d?d?d
Output: None
Estimate time: 11 secs
Challenge: 4137b9bce6c0e826abf062cb677a1498 and Other 9 MD5
    Hash
hashcat -m 0 -a 3 hash.ebu ?l?l?l?l?l?l?l?l?l?l
Output: cambridgedw
Estimate time: 36 mins 24 secs
```

The attacker extracted the AuthInfo dataset features to analyze two cracked passwords. After analysis in Table A2, an eight-character lowercase letter in all user passwords has been discovered to be a password prefix. The attacker combines this prefix with previously analyzed password behaviors, and he conducts a brute force attack on the victim user’s other password hashes. Eight characters are fixed in this pattern, and the “cambridg” prefix is prepended to the attack for different passwords. Thus, the attacker compromised the majority of the passwords.

Table A2
Cracked Hash with AuthInfo Feature.

Feature	Cracked Hash 1	Cracked Hash 2
Password	cambridg1	cambridgedw
Length	9	11
Type of Char	dl	l
Source	1. Data Source	2. Data Source
Mask	llllllld	lllllllllll
Type of First Char	l	l
Type of End Char	d	l
Number of Lower Case	8	11
Number of Upper Case	0	0
Number of Special Char	0	0
Number of Decimal	1	0

Table A3
Attack times with different password length.

Character Set	Prefix	Password Length	Attack Time
lowercase and decimal	cambrige	17	over 10
lowercase and decimal	cambrige	18	2 months
lowercase and decimal	cambrige	19	8 months
lowercase and decimal	cambrige	20	57 years

The prefix information helped the attacker to crack 11-16 character lengths within ten hours. For this operation, the command and the output are given below.

```
Challenge: e0ce9da5d746df9b52d0d8a0fa0c96ec and Other 8 MD5
    Hash
hashcat -a 3 -m 0 -i --increment-min=11 -1 ?d?l hash.ebu
    cambridg?1?1?1?1?1?1?1?1?1?1?1?1?1
Output: cambridged1, cambridged0w, cambridged0wq,
    cambridge12177, cambridgeasdaq, cambridge123qqq,
    cambridgeazasd1d
Estimate time: 10 hour 20 mins, 32 secs
```

After the attack, the mask is examined as a password pattern, and it is discovered that the victim uses variable-length numbers and lowercase letters after using a lowercase phrase with eight characters such as LLLLLLD, LLLLLLLLDLL, and LLLLLLLDDDLLL. When the cracked hashes are re-evaluated with AuthInfo features, the attacker detects that passwords longer than 9-characters have a “cambridge” string prefix. If the attacker had used his new prefix in the previous attack, the brute force attack would have lasted close to 20 minutes instead of 10 hours. The required time to attack longer passwords is given in Table A3. For this operation, the command and the output are given below.

```
Challenge: aaf4b0bb8dca5d91592e25ceafb62e57 and Other MD5 Hash
hashcat -a 3 -m 0 -i --increment-min=16 -1 ?d?l hash.ebu
    cambridge?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1
Output: cambridgeazasd1d
Estimate time: 18 mins, 29 secs
```

The attacker broke nine out of ten MD5 hashes by utilizing the characteristics and analysis of the authInfo dataset and shown in Fig. A.11. Next, the attacker tried all the passwords in the attack space, using the statistics and the information he learned from the passwords he broke. Finally, evaluating the possibility that the last password is a password other than the password behavior expanded the attack space with capital letters and attacked again. This last brute force attempt has failed since the character count has increased. A brute force attack involving all character sets also failed. Finally, the attacker tried to enlarge the brute force space with the “cambridge” prefix. None of the attack vectors implemented have been successful. The failed attack commands are given below.

```
Challenge: eeb266e017bab6512e843f96dfb1c6d1
hashcat -a 3 -m 0 -i --increment-min=4 -1 ?d?l?u hash.ebu
    ?1?1?1?1?1?1?1?1?1?1?1
Output: None
hashcat -a 3 -m 0 -i --increment-min=4 -1 ?d?l?s hash.ebu
    ?1?1?1?1?1?1?1?1?1?1?1
Output: None
hashcat -a 3 -m 0 -i --increment-min=9 hash.ebu ?a?a?a?a?a?a?a
    ?a?a?a
Output: None
hashcat -a 3 -m 0 -i --increment-min=9 hash.ebu cambridge?a?a?
    a?a?a?a?a?a?a?a
```

Although the attacker offers a nine-character prefix for the victim’s longest password of 22 characters, the attacker cannot succeed because it exceeds the brute force limits of the hashcat program. However, the calculated time is more than ten years (Next Bigbang) when the attack starts with the nearest 21 digits mask. Hash, run commands, and outputs are given below.

```
Challenge: eeb266e017bab6512e843f96dfb1c6d1
hashcat -m 0 -a 3 -1 ?d?l hash.ebu cambridge
    ?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1
output: none
hashcat -m 0 -a 3 -1 ?d?l hash.ebu cambridge
    ?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1?1
output: none
estimate time: more than ten years (Next Bigbang)
```

Additionally, a dictionary can be created by predicting the user’s next password, using the analysis information obtained, using the properties specified in the data set. However, because users prefer password lengths ranging from 9 to 22 characters, creating a dictionary is quite expensive in processing power and disk space.

References

Awad, M., Al-Qudah, Z., Idwan, S., Jallad, A.H., 2016. Password security: Password behavior analysis at a small university. In: 2016 5th International Conference on Electronic Devices, Systems and Applications (ICEDSA). IEEE, pp. 1–4.

Barkadehi, M.H., Nilashi, M., Ibrahim, O., Fardi, A.Z., Samad, S., 2018. Authentication systems: A literature review and classification. Telematics and Informatics 35 (5), 1491–1511.

Belding, G., 2018. What are honeywords? password protection for database breaches. https://resources.infosecinstitute.com/topic/what-are-honeywords-password-protection-for-database-breaches/.

Choong, Y.-Y., Theofanos, M.F., Liu, H.-K., 2014. United States Federal Employees' Password Management Behaviors: A Department of Commerce Case Study. US Department of Commerce, National Institute of Standards and Technology.

Erguler, I., 2015. Achieving flatness: Selecting the honeywords from existing user passwords. IEEE Transactions on Dependable and Secure Computing 13 (2), 284–295.

Grobler, M., Chamikara, M., Abbott, J., Jeong, J.J., Nepal, S., Paris, C., 2020. The importance of social identity on password formulations. Personal and Ubiquitous Computing 1–15.

GVEN, E. Y., 2021. https://github.com/istec-iuc/AuthInfo-Dataset.

Hu, G., 2017. On password strength: a survey and analysis. In: International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing. Springer, pp. 165–186.

Hunt, T., 2021. https://haveibeenpwned.com/About.

Jayakrishnan, G.C., Sirigireddy, G.R., Vaddepalli, S., Banahatti, V., Lodha, S.P., Pandit, S.S., 2020. Passworld: A serious game to promote password awareness and diversity in an enterprise. In: Sixteenth Symposium on Usable Privacy and Security ({SOUPS} 2020), pp. 1–18.

Kävrestad, J., Zaxmy, J., Nohlberg, M., 2019. Analysing the usage of character groups and keyboard patterns in password usage. In: Human Aspects of Information Security & Assurance (HAISA 2019) International Symposium on Human Aspects of Information Security & Assurance (HAISA 2019), Nicosia, Cyprus, July 15-17, 2019. University of Plymouth Press, pp. 155–165.
Malderle, T., Wübbeling, M., Knauer, S., Sykosch, A., Meier, M., 2018. Gathering and analyzing identity leaks for a proactive warning of affected users. In: Proceedings of the 15th ACM international conference on computing frontiers, pp. 208–211.
Maoneke, P.B., Flowerday, S., Isabirye, N., 2020. Evaluating the strength of a multilingual passphrase policy. Computers & Security 92, 101746.
Maschler, F., Niephaus, F., Risch, J., 2017. Real or fake? large-scale validation of identity leaks. Informatik 2017.
Murphy, D.S., 2018. Analysis of User Response to Complexity in Password Composition Policies in US Healthcare Organizations. Northcentral University.
Pagar, V.R., Pise, R.G., 2017. Strengthening password security through honeyword and honeyencryption technique. In: 2017 International Conference on Trends in Electronics and Informatics (ICEI), pp. 827–831. doi:10.1109/ICOEI.2017.8300819.
Shay, R., Bauer, L., Christin, N., Cranor, L.F., Forget, A., Komanduri, S., Mazurek, M.L., Melicher, W., Segreti, S.M., Ur, B., 2015. A spoonful of sugar? the impact of guidance and feedback on password-creation behavior. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, pp. 2903–2912.
Shay, R., Komanduri, S., Kelley, P.G., Leon, P.G., Mazurek, M.L., Bauer, L., Christin, N., Cranor, L.F., 2010. Encountering stronger password requirements: user attitudes and behaviors. In: Proceedings of the sixth symposium on usable privacy and security, pp. 1–20.
Thite, M.V., Nighot, M., 2021. Honeyword for security: A review. International Journal 6 (5).
Wang, D., He, D., Cheng, H., Wang, P., 2016. fuzzypsm: A new password strength meter using fuzzy probabilistic context-free grammars. In: 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). IEEE, pp. 595–606.
Wang, D., Wang, P., He, D., Tian, Y., 2019. Birthday, name and bifacial-security: understanding passwords of chinese web users. In: 28th {USENIX} Security Symposium ({USENIX} Security 19), pp. 1537–1555.
Wash, R., Rader, E., Berman, R., Wellmer, Z., 2016. Understanding password choices: How frequently entered passwords are re-used across websites. In: Twelfth Symposium on Usable Privacy and Security ({SOUPS} 2016), pp. 175–188.
Wheeler, D.L., 2016. zxcvbn: Low-budget password strength estimation. In: 25th {USENIX} Security Symposium ({USENIX} Security 16), pp. 157–173.
Wu, T., Yang, Y., Wang, C., Wang, R., 2019. Study on massive-scale slow-hash recovery using unified probabilistic context-free grammar and symmetrical collaborative prioritization with parallel machines. Symmetry 11 (4), 450.
Yang, W., Li, N., Molloy, I.M., Park, Y., Chari, S.N., 2016. Comparing password ranking algorithms on real-world password datasets. In: European Symposium on Research in Computer Security. Springer, pp. 69–90.
Yank, K., 2021. https://breachalarm.com/sources/.

Ebu Yusuf Güven has received his bachelor degree from Istanbul University Computer Engineering in Turkey. At Fatih Sultan Mehmet University, he completed his master's degree with a thesis titled "Cyber Attack Detection and Prevention Methods for Edge Computing". He is currently a PhD student at Istanbul University-Cerrahpasa, and his thesis is entitled " Development of a New Scan Model for Cyber Threat Intelligence". He works as research assistant at Istanbul University-Cerrahpasa and researcher at the IoT Security Test and Evaluation Center (ISTEC). He has strong interests in cyber security and related fields.

Ali Boyaci received the B.S. and M.Sc. degrees in computer science from Istanbul University, Istanbul, Turkey, in 2007 and 2010, respectively, and the Ph.D. degree from the Yildiz Technical University, Istanbul, Turkey, in 2015. He worked as a software engineer at Nortel Networks and project leader at Huawei from 2007 to 2012. Currently, he is an Assistant Professor with the Department of Computer Engineering, Istanbul Commerce University, Istanbul. Ali Boyaci's current research interests include computer networks, vehicular networks, operating system, programming languages, software development and embedded systems.

Muhammed Ali Aydin obtained his B.S. degree in computer engineering from Istanbul University in Istanbul, Turkey in 2001. He completed his MSc degree in computer engineering from Istanbul Technical University, Istanbul, Turkey in 2005. He received his Ph.D. degree in computer engineering from Istanbul University, Istanbul, Turkey in 2009. He was a Postdoctoral Research Associate with the Department of RST, Telecom SudParis, Paris, France, from 2010 to 2011. He has been working as an Assistant Professor in Istanbul University-Cerrahpasa Department of Computer Engineering since 2009. He is the Vice Dean of Engineering Faculty and Head of Cyber Security Department since 2016. His research interests include optical networks, network security, information security and cryptography.