Work-in-progress: Spurious Credentials in Breach Compilations

Lucas Stephens
Oregon State University
stephluc@oregonstate.edu

Jacob Porter
Oregon State University
portjaco@oregonstate.edu

Zane Ma
Oregon State University
zane.ma@oregonstate.edu

***Abstract*—Compilations of credential breaches underlie and inform a substantial body of password-related research, including password guessing/strength models, honeywords, credential breach checking, etc. However, these breach compilations often have murky provenance, and thus questionable fidelity. This study examines *spurious credentials*—credentials that don't represent user password behavior—in commonly used credential breach compilations. Our work finds several types of spurious credentials overlooked by all prior research, including invalid credentials and outlier, automated credentials. Confirming spurious credentials is challenging due to the lack of large-scale ground truth data. Thus, we perform an in-depth case study of the largest instance of spurious credentials and find compelling evidence of credential manipulation and automatic generation. In total, our analysis finds that spurious credentials comprise at least 2.9–5.4% of prior breach compilations, pointing to the need for further research on validating credential compilation datasets.**

I. Introduction

Credential data breaches have become a common occurrence, with millions or even billions of users affected over the past decade. While these breaches are often used maliciously (e.g., credential stuffing, account hijacking), they also provide an opportunity for defenders and security researchers to better understand the nature of user credentials, the scope of potential harms, and even discover new credential-related attacks. Ultimately, this research informs downstream credential policies (e.g., password composition, password expiration, etc.) that influence billions of web users.

Credential data used for research is often sourced from breach compilations that aggregate credentials across dozens or hundreds of independent breaches. These compilations have unknown origins, and thus unknown integrity. Despite this uncertainty, dozens of prior works have utilized these data sources with little-to-no data cleaning, raising questions about hidden biases and limitations of prior findings.

This paper takes a first look at the presence of *spurious credentials* in two of the most widely used credential breach compilations: the 2017 compilation published by 4iQ [5] and the Compilation Of Many Breaches (COMB) [11] released in 2021. Spurious credentials encompass both invalid credentials (e.g., emails containing multiple @) and large-scale outlier credentials that were likely generated by automated means (e.g., 1% of all 4iQ passwords start with `fbobh_`); both fail to reflect standard user behavior that is the implicit target of password-focused research. Spurious credentials have many root causes including improper breach data processing (e.g., listing password hash instead of plaintext), corrupted data, sybil account creation, post-compilation injection for inflating credential numbers, etc.

Current practice for validating the fidelity of breach compilations consists of spot-checking true positives [8], and there is limited consideration of spurious credentials (false positives). Research using these data sources only performs minor data cleaning that filters out short/long passwords, hexadecimal strings (likely password hashes), and non-ASCII characters. However, we argue that these simple filters fail to remove a majority of spurious credentials. We apply new data cleaning techniques to remove syntactically invalid emails, statistical outlier password/email distributions, and a particular high volume spurious credential, `fbobh_*`. By taking a conservative approach that prioritizes precision over recall, we identify an additional 41.6M (2.9%) and 175.3M (5.4%) spurious credentials in the 4iQ and COMB datasets, respectively.

The lack of at-scale ground truth data makes it difficult to determine whether credentials are actually spurious. Many of the identified credentials rely on certain assumptions about the behavior of a real user (i.e., real users do not repeat email patterns thousands of times), which can mislabel the credentials belonging to some extreme outlier users. However, despite these limitations, we believe our findings still represent an improvement over existing credential cleaning techniques. Furthermore, we perform an in-depth case study on the most significant case of spurious credentials and find strong evidence of credential injection and manipulation across both breach compilations.

Ultimately, this work provides a first look at the fidelity of the data underlying a substantial body of credential research. Our research highlights the need to better validate these important datasets and provides tools and directions for future investigation.

In summary, our work makes the following contributions:

- Comparison of prior cleaning methods for credential breach compilations, detecting varying degrees of data hygiene.
- New syntactic and statistical methods for data cleaning that identify a lower-bound 2.9–5.4% additional

Workshop on Measurements, Attacks, and Defenses for the Web (MADWeb) 2026
27 February 2026, San Diego, CA, USA
ISBN 978-1-970672-06-0
https://dx.doi.org/10.14722/madweb.2026.23018
https://www.ndss-symposium.org/

spurious credentials.

- Detailed case-study that suggests many forms of data manipulation for a well-attributed set of spurious credentials.
- We open-source[1] our credential cleaning code for future research utilizing breach compilation data.

II. Background and Related Works

A. Credentials

A credential consists of a user identifier and a password. While user identifiers can sometimes be arbitrary strings (depending on the service), all of the user identifiers found in credential breach compilations are email addresses. Email addresses are not case sensitive, but passwords are. We further break down the email address into [username]@[domain] when describing the portions of the email that is used for identifying spurious credentials. The email "username" is technically called the "local-part"; however, we use the former, more common terminology throughout this work.

B. Breach Compilations

The 2017 breach compilation [5] discovered by 4iQ (now Constella Intelligence) is one of the most widely used credential datasets and contains 1.4 billion plaintext credentials. It contains email address and password tuples organized by email address prefix. The data is distributed with a `imported.log` file that lists 252 filenames, file sizes, and file hashes that supposedly reflect the sources of the aggregated credentials. About half of the filenames are generic, while the other half have some indication of a specific compromised web service. There is no linkage between individual credentials and breach sources and no other information about the potential provenance of the credentials.

The Compilation of Many Breaches (COMB) [11] appeared in 2021 on a hacking forum and contains 3.3 billion credentials, more than double the size of 4iQ. COMB was reported to contain more recent data breaches, but unlike 4iQ, COMB does not contain any indication of what breaches the credentials are purportedly sourced from. We investigated COMB because it contains substantially more credentials than 4iQ, is utilized by several recent works, and has significant overlap with the 4iQ breach. The overlapping data presents an opportunity to compare the two datasets and examine their inter-compilation consistency.

C. Prior cleaning efforts

Dozens of prior research works have utilized the 4iQ or COMB datasets. To find a meaningful sample, we searched for all works utilizing this data in top security conferences from the last six years (2019–2025), which yielded twelve total works. Table I indicates the cleaning methods for these prior works, which vary greatly (0–59.4M for 4iQ) in the number of credentials removed during data cleaning.

Many of the twelve identified papers shared similar techniques: nine enforced a length requirement on the email or password, and seven removed lines with non-ASCII characters. Several papers also employed unique techniques. For example, one paper removed credentials containing spaces [21], and another removed emails without the "@" character [10]. We tried to replicate the data cleaning results of prior work, but we were unable to reproduce them exactly (see Appendix A). Half of our replications resulted in fewer than the reported removals, and the other half more. In one instance, two papers by the same authors used different cleaning methods but reported the same number of removals [19], [17]. One of our replications exceeded the reported number, while the other fell short.

Prior Work	**Cleaning Filters(s)**	**Dataset**	**Creds. Removed**
2019 S&P [12] 2022 Sec. [13]	PW < 4 || > 30 chars. PW ≥ 20 hex substring PW non-ASCII chars > 497 PW per email PW post-cleaning dedup.	4iQ	31.89M
2019 CCS [10]	PW non-ASCII, > 30 chars. > 1000 PW per email	4iQ	19.24M
2021 Sec. [14] 2024 S&P [15]	*No cleaning*	4iQ	0
2021 CCS [16]	Invalid emails PW < 3 || > 31 chars. > 100 PW per email	4iQ	24.74M
2022 Sec. [4]	PW < 6 || > 30 chars. PW post-cleaning dedup.	4iQ	50.78M
2023 Sec. [20]	PW non-ASCII, > 32 chars. PW ≥ 20 hex substring	4iQ	28.99M
2024 Sec. [21]	PW < 4 || > 30 chars. PW ≥ 20 hex substring Has spaces, non-ASCII chars. Additional steps...	4iQ	27.99M
2023 Sec. [17]	PW < 1 || ≥ 30 chars. PW Non-ASCII chars Email missing @	4iQ COMB	59.42M 115.4M
2024 Sec. [19]	PW ≥ 30 chars. PW Non-ASCII chars Missing or invalid email	4iQ COMB	18.78M 38.41M
2025 Sec. [22]	PW non-ASCII, > 30 chars.	COMB	25.32M
This work	Syntactic (Section III-A)	4iQ COMB	50.26M 42.48M
This work	Password Outliers (Section III-B)	4iQ COMB	7.46M 49.81M
This work	Email Outliers (Section III-C)	4iQ COMB	17.41M 93.83M
This work	FBOBH (Section III-D)	4iQ COMB	14.22M 14.25M
Raw data	*No cleaning*	4iQ COMB	*1.4B total* *3.3B total*
Aggregate Prior Work	Non-ASCII chars Invalid emails PW < 4 || ≥ 30 chars PW ≥ 20 hex substring > 100 PW per email	4iQ COMB	40.9M 152.8M
This work	Union(New methods)	4iQ COMB	84.95M 117.07M
Total	Union(Standard prior + new)	4iQ COMB	118.98M 254.85M

TABLE I: Credential Dataset Cleaning

[1]https://github.com/ASTROLAB-OSU/Data-Cleaning-Scripts

We hypothesize that some of these inconsistencies stem from incomplete methodological descriptions. For example, some papers cite related work when describing their cleaning rules, yet they do not specify which cleaning rules were actually adopted. Even when cleaning procedures appeared to be described in full, our findings differed. Another possible source of variation is under-specified parsing of breach compilation data. In our study, we assume the expected credential format is either "email:password" or "email;password". However, the data in both 4iQ and COMB sometimes deviates from this format, often appearing corrupted or completely unstructured. Furthermore, encoding differences may also contribute to discrepancies. While UTF-8 is commonly used, many 4iQ and COMB files contain invalid UTF-8 values, and prior work does not describe how such scenarios are handled.

To build a representative baseline of prior cleaning efforts, we compiled a list of methods that were shared by at least two other papers (Table I). The one exception is that we applied the non-ASCII filter to the full credential, which is more restrictive than most prior works that only check for ASCII characters in the password. Our baseline aggregate cleaning removed 40.9M (2.92%) credentials from 4iQ and 152.8M (4.66%) from COMB.

Filtering Rule	**4iQ**	**COMB**	**Synthetic example**
Non-ASCII Characters	11.09M	15.62M	asg°43:"»¬'
PW Length $\leq$ 4 or $\geq$ 30	15.37M	36.39M	(email):89
PW Hex Substring $\geq$ 20	3.09M	4.74M	(username): 9391c60dca3a255baa70
Invalid Emails	3.90M	10.89M	32179@sca@yahoo.com: (password)
PW per Email > 100	12.26M	93.93M	sai@de.ru:biene1, sai@de.ru:birge01, sai@de.ru:bjqctu6, ...

TABLE II: Identified Credentials for Standard Prior Work

Even after reproducing prior cleaning methods, manual validation reveals a substantial number of likely auto-generated credentials. These types of methods are useful for ensuring input validation; however, they are not complex enough to confidently remove spurious credentials. Many uses of these data breaches rely on the assumption that the credentials were determined and evaluated by real users. Therefore, we propose new methodologies aimed at more accurately identifying spurious credentials.

III. Finding Spurious Credentials

To better identify and clean spurious credentials beyond prior work, we implemented four categories of filters: syntactic, password outliers, email outliers, and `fbobh_*`. We define the expected format for a valid credential in a data breach as "<email>[:/;]<password>(newline/tab/carriage return)" (e.g. "fake@cred.com;fa:ke;pass`\t\r\t\n`"). Not all lines in both the 4iQ and COMB datasets follow this format, so we naturally count lines that break this expectation as invalid but towards the total credential count.

A. Syntactic

Duplicate Credentials: Contrary to prior report [12], we found that the 4iQ dataset contains duplicate credentials, which we removed. In total, this filter identifies 6.59M (0.47%) credentials in the 4iQ and 0 (0%) in COMB.

TLD Check: Using a list of IANA recognized TLDs including six removed TLDs (5 defunct ccTLDs and one for NATO), we filtered out emails that do not use a valid TLD. These email addresses are not and have never been valid. While some services may allow account creation without email verification, we believe accounts with invalid TLD emails do not represent typical users. Concretely, we observed emails with no domain, entirely made-up TLD, or looked like a valid TLD that was missing a ".". In total, this filter identifies 6.86M (0.49%) credentials in 4iQ and 37.78M (1.15%) in COMB.

Email Length: Email lengths throughout both datasets vary greatly 4 5. The RFC email standards define the maximum local and domain lengths of an email to be $\leq$ 64 and $\leq$ 255 characters, respectively, while the entire length must be $\leq$ 254. This loose requirement on length is not maintained in practice. We find the minimum length of a practical email to be 6 total characters (e.g. `a@a.ac`). In total, this filter identifies 0.24M (<1%) credentials in 4iQ and 0.1M (<1%) in COMB.

B. Password Outliers

The password outlier filters preprocess all passwords in the dataset to find irregular patterns that highlight likely spurious credentials IV. Our hypothesis was that spurious passwords would have statistically outlying distributions when compared to legitimate passwords, therein providing a way to target and remove spurious credentials. We developed two techniques: Follow-on Ratio and Follow-on Distribution.

For both techniques, we insert each password into a trie that stores each character of a password as a node and counts the number of passwords ending at each node. With this trie, we can efficiently compute the number of occurrences for any password p and the passwords for which it is a prefix. For a given password p, we define the *standalone* count as the number of occurrences of p. We define the *follow-on* count as the sum of the occurrences for all children (not just direct children) of p in the trie, which is effectively the number of passwords beginning with p and containing at least one additional character.

Follow-on Ratio: Follow-on Ratio is used to identify passwords that have a significant drop between the standalone count and the follow-on count. This is a useful metric because common user passwords often have many follow-on variations that add numbers or symbols to the end of the password. On the other hand, consider the case of the high-entropy password: `s891kn4cjb10cq3`. Imagine if there are 60,000 standalone occurrences of this prefix with 0 follow-on occurrences (i.e. 0 passwords with a prefix including the high-entropy password and at least one extra character on the end), implying that 60,000 accounts somehow managed to share the exact same random-looking password. This example credential is highly unlikely to occur naturally and would be considered a spurious credential.

To determine the occurrence threshold required to identify these anomalies, we looked at the distribution of standalone and follow-on counts (Figure 1). For both the 4iQ and COMB datasets, we performed manual inspection to determine a piecewise-function to preserve all of the assumed non-spurious prefixes (Equation 1), with the shaded region including prefixes/passwords we remove. We export a list of these spurious prefixes in pre-processing for the filter to identify during cleaning. In total, the Follow-on Ratio filter identifies 6.66M (0.48%) credentials in 4iQ and 42.25M (1.29%) in COMB.

$$\text{threshold}(x) = \begin{cases} 1 & \text{if } 3000 \leq x \leq 5000 \\ \frac{x}{500} - 10 & \text{if } 5000 < x \leq 20000 \\ \frac{x}{120} - 135 & \text{if } x > 20000 \\ \text{ignore} & \text{if } x < 3000 \end{cases} \tag{1}$$

Follow-on Distribution: Follow-on Distribution is used to identify credentials similarly to Follow-on Ratio, but with a focus on the distribution of the immediate next (i.e., follow-on) character of the password prefixes themselves. We compare this distribution for each prefix to a baseline calculated from all prefixes. If a prefix has a character with a follow-on occurrence far outside what would be considered normal, it will be flagged. For example, the prefix "12345" would likely have a high follow-on occurrence of "6" as it follows a pattern, but likely a low follow-on occurrence of "j". Unnatural patterns with high follow-on occurrence are flagged for manual analysis.

First, we generate a list of all prefixes with more than 50,000 occurrences. We decided on this threshold as a balance between our manual analysis capabilities and overall data coverage. With this list of prefixes, we determined the average follow-on occurrence rate for each character and flagged any follow-on characters outside of the 5th and 95th percentages.

Second, we manually analyzed this list of prefixes to determine if the follow-on distributions seem like legitimate user behavior, or not. To get an objective stance on the prefixes, we had two experts independently review the list and mark prefixes as true positive or not. Upon completion of the review, conflicting findings would be discussed until a mutual decision was made to include or exclude the prefix in the final cleaning. Through this method, we determined 18 prefixes and 10 full passwords to filter out spurious credentials in 4iQ. The process was repeated for COMB where we found 67 prefixes and 27 full passwords. In total, the Follow-on Distribution filter identifies 1.55M (0.11%) credentials in 4iQ and 30.94M (0.94%) in COMB.

C. *Email Outliers*

The email outliers filters focus on trends via the emails of credentials when sorted IV.

Sequential Usernames: When manually reviewing the sorted data, we found extensive stretches of usernames followed by an incrementing number, all with the same password (e.g. 0000-0100@gmail.com:pass, 0000-0101@gmail.com:pass, 0000-0102@gmail.com:pass). This behavior can be explained when there are only a couple of occurrences; however, exceedingly long patterns likely indicate spurious origin that could skew analysis. We remove any 100 or more consecutive credentials with usernames containing an incrementing number that share the same password. In total, this filter identifies 14.25M (1.02%) credentials in 4iQ and 76.35M (2.33%) in COMB.

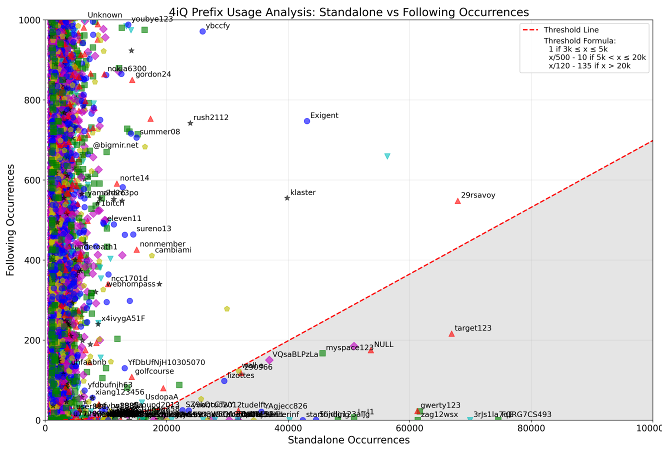

Fig. 1: **Follow-on Ratio Prefixes**: Graph indicating our conservative, manually-verified decision boundary for FOR outliers.

Email-Chains: During manual analysis of the sorted datasets, we noticed that specific email-domains were frequently grouped together with the same exact password; forming these "blocks" throughout. This felt like it could be a tell of potentially injected data, so we decided to create a filter to identify them.

Our Email-Chains filter looks for sequential credentials in the sorted dataset that share the same password and domain section of the email. A "chain" is the connection from domain-A to domain-B (e.g. line1@yahoo.de:pass -> line2@mail.ru:pass is one chain, line2@mail.ru:pass->line3@web.de:pass is another chain). Repeated chains increment their respective count to develop a distribution of all "chains" throughout the dataset. The filter then checks every domain to see if any of their chained domain(s) occurred over 50% of the time. This leaves us with a group of chains that we can use on a second pass of the dataset to remove credentials that share the same username and password. Often, credentials following this pattern will have an intermediary credential that does not match the general chain identified. This is due to the dataset being sorted by email, so these cases are still counted, and the chain is removed without removing the intermediary credential. In total, the Email-Chains filter identifies 14.25M (1.02%) credentials in 4iQ and 18.08M (0.55%) in COMB.

D. `fbobh_*`

This filter was developed due to some irregular findings during analysis of the 4iQ dataset. We identified a seemingly spurious prefix of passwords that was evading prior methods: `fbobh_` followed by 4 random characters. We were unable to reliably identify it with the Follow-on Distribution filter due to our limit on standalone occurrences, so we created this filter to target them specifically. When a password has the prefix `fbobh_` it is flagged. In total, the FBOB filter identifies 14.22M (1.02%) credentials in 4iQ and 14.25M (0.43%) in COMB.

Filtering Rule	4iQ	COMB	Example
Duplicate Credentials	42.13M	0	a@mail.ru:pass $\times$ 2
TLD Check	6.86M	37.79M	d91@couk:pass
Email $\leq 5 \mid\mid \geq 255$	0.24M	0.01M	5@a.r:pass
All Combined	*48.84M*	*37.79M*	–

TABLE III: Syntactic Filters

Filtering Rule	4iQ	COMB
Sequential Usernames	3.18M	76.35M
Follow-on Ratio	6.66M	42.25M
Follow-on Distribution	1.55M	30.94M
Email-Chains	14.25M	18.08M
FBOBH	14.22M	14.25M
All Combined	*39.05M*	*156.11M*

TABLE IV: Password + Email + fbobh_* Filters

IV. Results

All new filters are designed to target a unique aspect of observed spurious credentials. Figure 2 shows the overlap of credentials identified by each filter to ensure each is necessary. Password outliers show the most overlap by far and still identify 50.4% unique credentials. Prior work filters show some additional overlap with other filters, which is to be expected from very obvious spurious credentials that likely match multiple syntactic filters. The rest of the filters show that they identify a very unique group, properly targeting new spurious credentials.

Overall, we are confident that our filters are able to detect and remove a lower-bound of spurious credentials; however, we also believe there are more ways of identifying spurious credentials that are not explained in this paper. For example, the distribution of email lengths for both 4iQ (Figure 4) and COMB (Figure 5) show unexpected outliers. Ultimately, there is more work to be done to identify additional outliers in these credential breach datasets that influence many previous and future works in the privacy fields.

V. Case Study: fbobh_*

The single most egregious case of spurious credentials is passwords beginning with `fbobh_`, which comprised 14.2M (1.0%) and 14.3M (0.43%) of the total credentials found in 4iQ and COMB, respectively. Not only are these passwords prevalent, they also exhibit strong non-human properties, since they are consistently *rigid*—nearly all (99.8%) instances of `fbobh_`-prefixed passwords contain a suffix of exactly four characters (e.g., `fbobh_o30k`). The four-character suffix is uniformly *random* across alphabetic characters, symbols, and digits with the exception of `9`, as shown in Figure 3. We define a *standard* `fbobh_` credential as one that has a random four character suffix. If these passwords were generated by humans, we would expect deviations in suffix length and non-random distribution of characters, and we wouldn't expect millions of users to start their passwords with the semantically-meaningless `fbobh_` prefix to begin with! Ultimately, we treat all `fbobh_` passwords as spurious credentials.

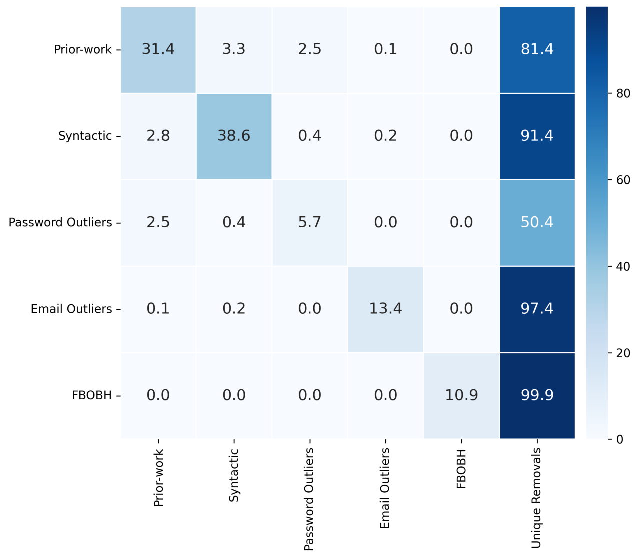

Fig. 2: Cleaning Method Overlap

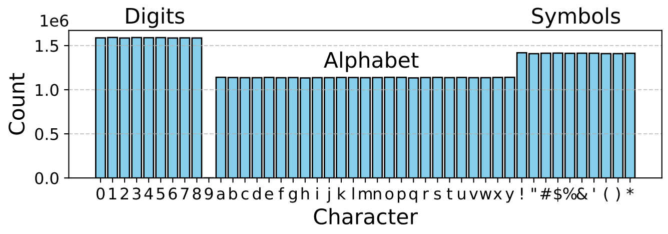

Fig. 3: **`fbobh_*` suffix distribution**: Separate, uniform distributions across digits (excluding 9), alphabet, and symbol characters.

A. *Investigating Origins*

To better attribute the origins of `fbobh_` credentials, we utilized the popular Have I Been Pwned (HIBP) service [2] that collects breach data and allows users to look up which breaches their email addresses were found in. As of December 2025, there are 927 breaches included in HIBP, spanning from July 2007 to November 2025. Due to rate limiting, we sampled one thousand random `fbobh_` domains and looked them up in HIBP to understand what specific breaches, if any, they were associated with. We found that all email addresses associated with `fbobh_` passwords across both 4iQ and COMB datasets had one breach in common: the MySpace data breach in 2008 [1], and that 294 (29.4%) of the sampled addresses were only associated with the MySpace breach. This evidence, along with online forum comments [7], [3], leads us to believe that the anomalous `fbobh_` credentials all originated from the MySpace breach in 2008.

B. *Comparing 4iQ and COMB*

Because of the strong attribution of `fbobh_` passwords, and the negligible likelihood of a random user manually choosing such a password, we can treat `fbobh_` as a tracker across different breach compilations. For this study, we compare 4iQ, which was released in 2017 and COMB, which was released in 2021. The distribution of fbobh_ credentials across these two breach compilations can provide insight into their differing credential inclusion criteria. In the simplest case, we might expect the MySpace breach fbobh_ credentials to be fully included in both datasets. However, we find that this is not the case, as many fbobh_ credentials are unique to 4iQ or COMB.

1) Intersection: Out of the 14.25M total fbobh_ credentials, 14.22M (99.8%) are found in both 4iQ and COMB. Upon closer inspection, the mapping of emails to passwords is many-to-many, indicating the presence of duplicate emails and duplicate passwords. 3,625 duplicate emails appear in the intersection, corresponding to 7,251 credentials (0.05%). In all cases except one[2], the duplicate emails have one *standard* fbobh_ password (i.e., contains a 4-character suffix drawn from the distribution in Figure 3, such as fbobh_abcd) and one *non-standard* truncated version of the password (e.g., fbobh_ab). This targeted truncation of fbobh_ passwords when duplicate emails are present hints at spurious injection and manipulation of the MySpace password data. After accounting for these duplicate email addresses, 70 non-standard fbobh_ passwords remain with unique email usernames. In every case, a previously accounted for non-standard credential is copied and then has the email provider altered to rambler.ru or yandex.ru. This represents another instance of fbobh_ credential manipulation that casts further doubt on the legitimacy of breach compilation data.

2) Unique to 4iQ: There are 11 standard fbobh_ passwords in 4iQ, each with a unique email address, that all are first attributed to the MySpace breach. Interestingly, these emails all contain at least one uppercase character, and all of the other emails associated with fbobh_ passwords are fully lower-cased. In fact, all eleven emails that are unique to 4iQ actually do appear in COMB in normalized lowercase form. This indicates that 1) 4iQ email addresses are not case-normalized, while COMB email addresses are, and 2) the COMB data contains a superset of the fbobh_ credentials found in 4iQ.

3) Unique to COMB: 30,211 fbobh_ credentials only appear in COMB, of which only 1,046 (3.5%) are standard fbobh_ passwords. Eleven instances are due to case-normalization of emails, described above. The remaining 1,035 standard fbobh_ passwords exhibit many varying forms of credential manipulation, including email provider rotation, password stuffing, numerically incrementing usernames, etc. These patterns of manipulation extend to the non-standard fbobh_ passwords as well. This points to a widespread injection of spurious fbobh_ credentials in the COMB dataset.

VI. Discussion

A. Impacts on Research

This study presents a lower-bound 2.9–5.4% rate of spurious credential inclusion in prior works, with the actual number being higher in nearly all cases since our representative baseline combines common elements from multiple prior works. The downstream impact of spurious credentials found in prior work remains an open question, and can vary greatly depending on how the credential data is used. Anecdotally, we find some password popularity lists [18] containing a handful of spurious credentials (e.g., dragon, which is part of the email chains) within the most common passwords. Future works could re-evaluate prior findings based on the newer, cleaned data to determine if there are out-sized (i.e., substantially greater than 5.4%) effects on certain findings. To aid future work using breach compilations, researchers can utilize the open source cleaning scripts we provide at: https://github.com/ASTROLAB-OSU/Data-Cleaning-Scripts. Additionally, there are opportunities to perform further cleaning techniques that are standard practice in other scientific fields [6].

B. Limitations and challenges

The definition of spurious credentials broadly encompasses any credentials that are not created by a user or a user-managed password manager, and we generally assume that spurious credentials should not be included in prior research. However, not all credential-based research is focused on user password behavior. For example, studies that use compilation breaches to test private breach notification protocols at scale might not care whether spurious credentials are included or not. That being said, we expect most credential breach research to primarily care about real, organic user credentials, and we ensure that all studies we examine in Table I utilize specific username/password properties that may be impacted by spurious credentials.

Verifying our findings is challenging due to the missing provenance information for compilation breaches. 4iQ provides some hints of its data sources, which could possibly be used to verify subsets of data fidelity. Broadly however, we believe it is important to develop new methods that can ethically find false positives and false negatives in compilation breaches or individual compilations themselves.

Understanding how and where spurious credentials found their way into breach compilations can be helpful for determining potential mitigations. There are two likely ways spurious credentials can make their way into breach compilation datasets. The first is lax credential management for individual services that are breached, allowing the addition of invalid or artificial bot-like credentials. The second is injection by the compiler/publisher of the breach compilation, who may inject spurious credentials to increase the perceived volume and financial value of the data. Distinguishing between these cases is challenging and would require data tied to the individual breaches underlying the compilations, which we did not have access to.

VII. Conclusion

Our study found 2.9–5.4% spurious credentials in two widely used breach compilation datasets. Combined with a deep-dive case study that points to specific instances of credential manipulation, our findings raise new questions about the fidelity of breach compilation data and introduce initial approaches and an open-source tool for cleaning out spurious credentials. Ultimately, we hope to spur further research in this area to more accurately understand real user password behavior.

[2]One email appears three times with one standard fbobh_ password and two non-standard passwords.

Ethics Considerations

Our study uses publicly available datasets that have been widely used in prior work (Table I); however the data itself (emails and passwords) is still private information. Because the individuals associated with this private data have not consented, and have not been able to consent to its use for research, the continued usage of this data raises ethical concerns. Guided by prior analysis of related ethical dilemmas [9], we take a consequentialist ethical approach that dictates that the benefits of the research must outweigh the harms. To reduce potential harms, we adopt best practices from prior works: 1) Our data is processed on access-controlled machines that are not accessible from the Internet, 2) we only present aggregated statistical information or specific examples with all identifying features removed, and 3) even though the data is public, we do not share it with anyone and will delete the data once our analysis is complete. Our work ultimately provides a new tool that enables more rigorous and accurate research on password credentials, which can improve related security products (e.g., password guessing / strength estimation models, breach notification, etc.) and password policies.

References

[1] https://web.archive.org/web/20160528094319/http://motherboard.vice.com/read/427-million-myspace-passwords-emails-data-breach.

[2] Have i been pwned. https://haveibeenpwned.com/.

[3] Alexander. "re: source of information for john's charset files". https://www.openwall.com/lists/john-users/2021/05/02/1#:~:text=ThenItriedusingHIBP,filefromthatcracks6.6%25., 2021.

[4] Marina Bohuk, Mazharul Islam, Suleman Ahmad, Michael Swift, Thomas Ristenpart, and Rahul Chatterjee. Gossamer: Securely measuring password-based logins. In *USENIX Security Symposium (Sec)*, 2022.

[5] Julio Casal. 1.4 billion clear text credentials discovered in a single database. https://medium.com/4iqdelvedeep/1-4-billion-clear-text-credentials-discovered-in-a-single-database-3131d0a1ae14, 2017.

[6] Xu Chu, Ihab F Ilyas, Sanjay Krishnan, and Jiannan Wang. Data cleaning: Overview and emerging challenges. In *International Conference on Management of Data*, 2016.

[7] hoffs. Myspace database dump layout? https://hackforums.net/showthread.php?tid=5338450, 2016.

[8] Troy Hunt. 2 billion email addresses were exposed, and we indexed them all in have i been pwned. https://www.troyhunt.com/2-billion-email-addresses-were-exposed-and-we-indexed-them-all-in-have-i-been-pwned/, 2025.

[9] Tadayoshi Kohno, Yasemin Acar, and Wulf Loh. Ethical frameworks and computer security trolley problems: Foundations for conversations. In *USENIX Security Symposium (Sec)*, 2023.

[10] Lucy Li, Bijeeta Pal, Junade Ali, Nick Sullivan, Rahul Chatterjee, and Thomas Ristenpart. Protocols for checking compromised credentials. In *ACM Conference on Computer and Communications Security (CCS)*, 2019.

[11] Bernard Meyer. COMB: largest breach of all time leaked online with 3.2 billion records. https://cybernews.com/news/largest-compilation-of-emails-and-passwords-leaked-free/, Feb 2021.

[12] Bijeet Pal, Daniel Tal, Rahul Chatterjee, and Thomas Ristenpart. Beyond credential stuffing: Password similarity models using neural networks. In *IEEE Symposium on Security and Privacy (S&P)*, 2019.

[13] Bijeeta Pal, Mazharul Islam, Marina Sanusi Bohuk, Nick Sullivan, Luke Valenta, Tara Whalen, Christopher Wood, Thomas Ristenpart, and Rahul Chatterjee. Might i get pwned: A second generation compromised credential checking service. In *USENIX Security Symposium (Sec)*, 2022.

[14] Dario Pasquini, Marco Cianfriglia, Giuseppe Ateniese, and Massimo Bernaschi. Reducing bias in modeling real-world password strength via deep learning and dynamic dictionaries. In *USENIX Security Symposium (Sec)*, 2021.

[15] Dario Pasquini, Danilo Francati, Giuseppe Ateniese, and Evgenios M. Kornaropoulos. Breach extraction attacks: Exposing and addressing the leakage in second generation compromised credential checking services. In *IEEE Symposium on Security and Privacy (S&P)*, 2024.

[16] Sena Sahin and Frank Li. Don't forget the stuffing! revisiting the security impact of typo-tolerant password authentication. In *ACM Conference on Computer and Communications Security (CCS)*, 2021.

[17] Ding Wang, Yunkai Zou, Yuan-An Xiao, Siqi Ma, and Xiaofeng Chen. Pass 2edit: A multi-step generative model for guessing edited passwords. In *USENIX Security Symposium (Sec)*, 2023.

[18] Wikipedia. List of the most common passwords.

[19] Kedong Xiu and Ding Wang. Pointerguess: Targeted password guessing model using pointer mechanism. In *USENIX Security Symposium (Sec)*, 2024.

[20] Ming Xu, Jitao Yu, Xinyi Zhang, Chuanwang Wang, Shenghao Zhang, Haoqi Wu, and Weili Han. Improving real-world password guessing attacks via bi-directional transformers. In *USENIX Security Symposium (Sec)*, 2023.

[21] Huang. Zhonghao, Lujo Bauer, and Michael K. Reiter. The impact of exposed passwords on honeyword efficacy second generation compromised credential checking services. In *USENIX Security Symposium (Sec)*, 2024.

[22] Yunkai Zou, Maoxiang An, and Ding Wang. Password guessing using large language models. In *USENIX Security Symposium (Sec)*, 2025.

Appendix A
Reproducibility Results

Prior Work	Dataset	Credentials Removed	Reported Removals	Difference
2019 IEEE S&P [12] 2022 Sec. [13]	4iQ	444.82M unique PW remaining	460.4M unique PW remaining	15.58M
2019 ACM CCS [10]	4iQ	19.24M	~80M	~60.76M
2021 Sec. [14] 2024 IEEE S&P [15]	4iQ	0	0	0
2021 ACM CCS [16]	4iQ	24.74M	2.8M	21.94M
2022 Sec. [4]	4iQ	447.28M unique PW remain 1.13B unique usernames remain	370M unique PW remain 1.12B unique usernames remain	77.28M ~10M
2023 Sec. [20]	4iQ	442.1M unique PW remaining	373.8M unique PW remaining	68.3M
2024 Sec. [21]	4iQ	443.79M unique PW remaining	563.7M unique PW remaining	119.91M
2023 Sec. [17]	4iQ COMB	59.42M 115.4M	19.05M 97.26M	40.37M 18.14M
2024 Sec. [19]	4iQ COMB	18.78M 38.41M	19.05M 97.26M	0.27M 58.85M
2025 Sec. [22]	COMB	25.32M	97.26M	71.94M

TABLE V: Previous Papers Actual vs. Reported Results

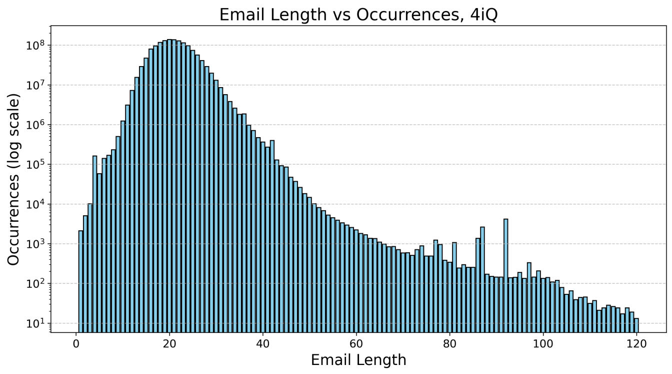

Fig. 4: Email Length Distribution, 4iQ

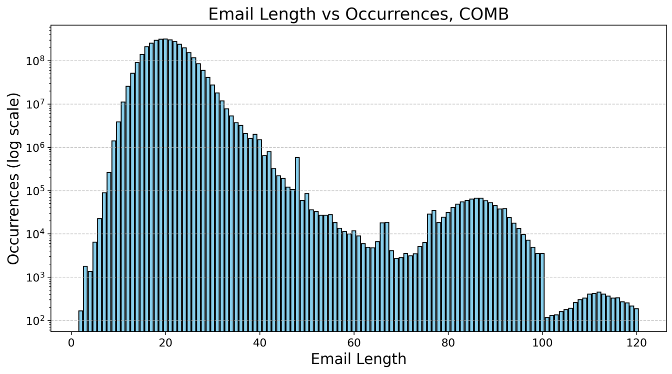

Fig. 5: Email Length Distribution, COMB