Password policies of most top websites fail to follow best practices

Kevin Lee Sten Sjöberg Arvind Narayanan

Department of Computer Science and Center for Information Technology Policy
Princeton University

This paper will be published in the *Proceedings of the Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*

Abstract

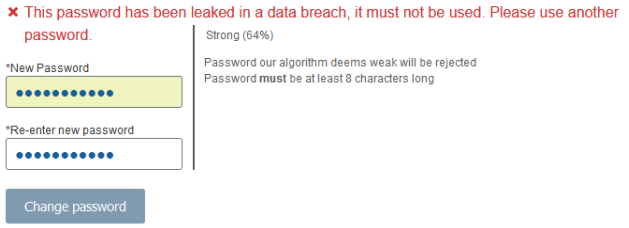
We examined the policies of 120 of the most popular websites for when a user creates a new password for their account. Despite well-established advice that has emerged from the research community, we found that only 13% of websites followed all relevant best practices in their password policies. Specifically, 75% of websites do not stop users from choosing the most common passwords—like "`abc123456`" and "`P@$$w0rd`", while 45% burden users by requiring specific character classes in their passwords for minimal security benefit. We found low adoption of password strength meters—a widely touted intervention to encourage stronger passwords, appearing on only 19% of websites. Even among those sites, we found nearly half misusing them to steer users to include certain character classes, and not for their intended purpose of encouraging freely-constructed strong passwords.

1 Introduction

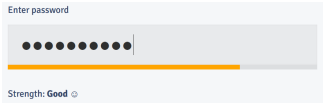
Passwords remain the most common means of authentication on the web, despite their shortcomings. According to industry estimates, close to half of data breaches involved authentication failures [13, 14]. As such, the need to use strong passwords remains unchanged [15]. To encourage this, websites mainly use three types of interventions during password creation: blocklists, password composition rules / policies (PCPs), and strength meters (Fig. 1). All three interventions have been extensively researched in the information security community.

Prior research has generally concluded that blocklists and strength meters—when configured correctly—lead users to create stronger passwords without significantly burdening them [3, 6, 16]. However, PCPs that require specific character-classes (i.e., lowercase, uppercase, digits, and symbols) are not recommended. That's because users fulfill requirements in predictable ways like capitalizing the first letter or placing a "!" at the end, negating the putative security benefits [17–19]. Additionally, character-class PCPs have consistently received poor usability ratings; in those same studies, users needed more attempts to create a compliant password and had difficulty recalling the password. Instead, websites should set only a minimum-length requirement while complementing it with a blocklist check or minimum-strength requirement [3].

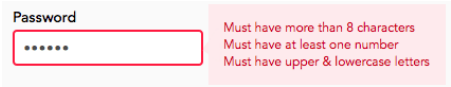
The research is clear; what is less clear is whether these best practices are actually being followed. There has been no comprehensive study to understand how online services guide their users in setting up passwords (although previous studies have looked at narrow aspects of this question [5, 20]). We aimed to fill this gap by examining password policies of 120



(a) A website preventing us from using a password ("`passer2009`") that was leaked in a data breach.



(b) An example of a password strength meter. Its colored bar and text feedback changes in response to the entered password.



(c) A *3class8* character-class PCP, which requires passwords be at least 8 characters in length with at least 1 lowercase, 1 uppercase, and 1 number.

Figure 1: Examples of the three interventions we studied: blocklists, PCPs, and password strength meters.

USENIX Symposium on Usable Privacy and Security (SOUPS) 2022.
August 7–9, 2022, Boston, MA, United States.

	Best practices from prior research	Our key findings
Blocklists (§ 3)	• Do check users' passwords against lists of leaked and easily-guessed passwords [1–4]. • Do reject the password if it appears on a blocklist, prompt the user to select a different password [1, 4].	• More than half (71 / 120) of websites do not check passwords at all, allowing all 40 of the most common passwords we tested (e.g., "12345678", "rockyou"). • 19 more websites block less than half of the most common passwords we tested.
Strength meters and min-strength reqs (§ 4)	• Do provide real-time password strength estimates [5–7]. • Do set minimum-strength requirements by estimating guessability (the number of guesses it would take for an adversary to crack the password) [3, 8–11].	• Only 23 / 120 websites used password strength meters. • Of those 23, 10 websites misuse meters as nudges toward character-class PCPs and do not incorporate any notion of guessability.
Composition policies (§ 5, § 6)	• Do not require specific character-classes; let users freely construct passwords [2, 3, 7, 12]. • NIST: Do set a minimum-length of at least 8 characters [1].	• 54 / 120 sites still use character-class PCPs. • We devised a new method to measure the security and usability of all 120 PCPs. Based on our method, we found that all PCPs performed poorly, none provided $\geq 60\%$ security and usability simultaneously.

Table 1: We contrast our key findings with established best practices for encouraging strong passwords.

of the most popular English-language websites in the world. By signing up for accounts and manually testing requirements for password creation, we discovered each website's blocklist strategy, PCP, and strength meter implementation (if any). We asked the following research questions:

1. Are websites preventing users from using the most common passwords? (§ 3)
2. Are websites using password strength meters to encourage strong passwords? (§ 4)
3. What PCPs are used by top websites? What are the security-usability tradeoffs of those PCPs? (§ 5, § 6)

We considered a website to be following best practices if it simultaneously satisfied the following security and usability criteria:

- **Security:**
  - Allowed 5 or fewer of the 40 most common leaked passwords and easiest-to-guess passwords (e.g., "12345678", "rockyou") we tried.
  - Required passwords be no shorter than 8 characters OR employed a password strength meter that accurately measured a password's resistance to being guessed by an adversary [7].
- **Usability:** Did not impose any character-class requirements.

We found that only 15 websites were following best practices. The remaining 105 / 120 either failed to adhere or explicitly flouted those recommendations in their policies, leaving users at risk for password compromise or frustrated from being unable to use a sufficiently strong password. We compare our key findings with the best practices for all three interventions in Table 1.

We further devised a method to measure the security and usability of PCPs using a large corpus of breached passwords. Past studies have typically examined a small number of different PCPs due to constraints with hiring participants, which motivated us to design a method that could scale to the large number of PCPs we examined. These studies have also systematically neglected to investigate PCPs with short minimum-length requirements, which we frequently found during our study (the following paragraph suggests a reason why previous studies may have excluded these PCPs). While we were able to analyze the PCPs of all 120 websites we visited, we note that our strategy has limitations and should be used to complement findings from previous user studies. We found that no PCP had more than 60% security and usability simultaneously. These results further call into question some of the recommendations on PCPs that have been taken at face value, without any evidence.

While there is broad consensus on best practices in the prior literature, it is sometimes unclear exactly where to draw the line. For instance, the National Institute of Standards and Technology (NIST) recommends an 8-character minimum-length requirement in its current version of *Digital Identity Guidelines*—a widely relied-upon resource by both practitioners and researchers [1]. Yet, that recommendation does not cite any research. Even though we performed a thorough literature search and failed to find any research that had investigated the usefulness of setting an 8-character minimum-length, we

decided to count that recommendation as a best practice. Here, we have used our best judgment in defining what constitutes best practice, erring on the side of being lenient. While our exact number might change if we change our definition of "best practices", our qualitative finding—that most websites are not following best practices—does not change.

Our findings reveal a disconnect between industry and the research community. Passwords have been heavily researched, yet few websites have implemented password policies that reflect the lessons learned. Researchers should make sure their findings have societal impact by engaging in outreach to website operators about their password practices.

2 Overview of password best practices

Websites mainly have three different ways to encourage users to create more secure passwords, as outlined by NIST [1]. Here we discuss previous research on the methods and their best practices.

2.1 Blocklists work, but need to be carefully configured

One simple way for websites to encourage more secure passwords is to keep a list of common insecure passwords (e.g., "`123456`", "`!QAZ1qaz`") and deny users from choosing passwords from that list (Fig. 1a). Prior research has found that password blocklists work. Kelley et. al (2012) created a blocklist of five billion passwords returned by a cracking algorithm created by Weir et. al (2009), and tested it in a subsequent user study [2, 21]. They found that users created passwords that were significantly harder-to-guess, compared to passwords created under four other widely-used but smaller blocklists. Shay et al. (2015) found that blocklists generally increase security without sacrificing password recall among users [7]. Habib et al. (2017) also supported using blocklists, and further recommended that websites also restrict users from submitting simple modifications to blocklisted passwords [16].

Blocklists may consist of common passwords gathered through different strategies, including commonly used passwords that have been exposed in data breaches and passwords that are likely to be guessed easily by password cracking tools. Websites may also have different approaches to checking passwords against the blocklist; for instance, some may perform exact matching while others strip out symbols before matching. While NIST recommends that websites block common passwords, it is neither prescriptive on which lists to use nor on the comparison method [1].

The National Cyber Security Centre (NCSC) provides more concrete guidance [4]. In collaboration with *Have I Been Pwned?* (HIBP)—an online service that allows users to check whether their credentials have been compromised in data breaches—the NCSC has released a list of the 100,000 most common passwords for websites to use as a blocklist (which we refer to as NCSC-HIBP-100k later on in the paper). NCSC guidance reasons that blocking the top 100,000 passwords prevents users from "making poor password choices, whilst not making it too difficult for them to choose one."

Tan et al. (2020) later investigated the security-usability tradeoffs of blocklist requirements and found that blocklists—while effective—can cause user frustration if not properly configured [3]. They recommend blocking passwords that appear in NCSC-HIBP-100k or blocking common passwords that appear in a corpus of 10 million leaked passwords, both of which we used in our experiments.

In this study, we empirically examine whether websites follow the best practices for blocklists established by prior work.

Compromised credential checking. In addition to blocking the most common passwords, some websites may employ compromised credential checking, which checks whether a username-password pair has been exposed in a previous breach [22, 23]. Websites can more accurately measure the risk of account compromise to the user because they additionally consider whether her full login credentials are already available to cybercriminals.

We did not test for compromised credential checking by websites in our study because it presents practical difficulties. Using other people's compromised credentials raises ethical concerns, whereas using our own compromised credentials may be unreliable due to the small sample size. While not listed by NIST as a best practice, some websites in our study may check for compromised credentials due to its known effectiveness.

2.2 Min-strength requirements and strength meters are both effective and user-friendly

A newer approach to encourage strong passwords is to set minimum-strength requirements. When a user submits a candidate password, the website estimates the strength for the submission, and if it is greater than the minimum threshold, the candidate password is accepted. A strength meter that updates in real-time is often shown to nudge users as they craft their passwords (Fig. 1b).

To measure strength, researchers recommend and typically use adversarial guessing—the number of guesses needed to crack a password (i.e., the guess number or guessability). Previously, strength was often modeled using Shannon entropy—a function of the length and number of character-classes present in a password, or its complexity. However, complexity has since been deprecated since it is not a good proxy for guessability (see Appendix F for further background).

Estimating password strength is difficult, especially considering that users expect near-instantaneous feedback when setting a password. Previous research has found that among the password-strength meters in use on the web, most actually measured complexity instead of guessability, and were actually inconsistent with one another (de Carnavalet and Mannan, 2014) [5]. There was an open-source implementation that were found to be reliable, however: `zxcvbn` outputs

accurate strength estimates through 10^6 guesses, the threshold for online attacks [24].

In 2017, Ur et al. designed a data-driven strength meter that estimates password strength using a client-side neural-network created in a prior study (Melicher et al., 2016) [10]. Their meter received positive feedback from participants in the following user study, and was accurate when compared with results from password cracking tools, which were used as ground truth [6]. Tan et al. (2020) later updated the meter to enforce blocklists and minimum-strength requirements, while also making the meter freely available to use [3]. They concluded that minimum-strength requirements are the best way to encourage strong passwords, and recommend setting the minimum-strength threshold to at least 10^6 to prevent online guessing attacks [3]. Since their password-strength meter directly estimates guessability—as opposed to PCPs indirectly using complexity as a measure of strength—websites need only set a minimum-guesses threshold instead of character-class requirements, such as 10^6 for online attacks and 10^{14} for offline attacks. They further highlight that the meter's underlying neural network can be "easily retrained to reflect changing patterns in passwords over time" and that its configurable integration with blocklists can penalize common passwords.

2.3 Character-class PCPs should not be used

To enforce the use of strong passwords, websites have employed password composition policies (PCPs). PCPs are rules which users must follow in creating their passwords. These rules most often include a minimum password length requirement along with character-class requirements (Fig. 1c). PCPs fall into two categories: ones with character-class requirements (which we'll refer to as "character-class PCPs" throughout the paper) and ones without (PCPs that only have a minimum length requirement, which we'll refer to as "minimum-length PCPs").

As a vestige of when password strength was modeled by Shannon entropy, character-class PCPs force users to create complex passwords.[1] While prior empirical research has found that passwords containing multiple character classes were generally more resilient to password-guessing attacks, employing character-class PCPs is a hardly ideal solution (Komanduri et al., 2011), (Kelley et al., 2012) [2, 12]. Character-class PCPs have poor usability; users have found it difficult to comply with the complex rules and to remember the password they have created. Moreover, character-class PCPs do not account for a significant subset of users, who respond predictably to comply with character-class requirements (e.g., adding numbers at the end, capitalizing the first letter). These behaviors reduce the benefits of adding complexity (Shay et al., 2010), (Weir et al., 2010), (Ur et al., 2015) [18, 19, 25]

As studies that have found that increasing minimum-length requirements while reducing character-class requirements can lead to strong passwords without decreased memorability, NIST has also updated its guidance to recommend websites remove character-class requirements (Kelley et al., 2012), (Shay et al., 2014) [2, 17]. It further recommended that websites require passwords be at least 8 characters long [1]. Tan et al. (2020) actually found that character-class PCPs do not make it harder for attackers using modern-day cracking tools anymore, since users now tend to incorporate multiple character-classes of their own accord. Still, they recommend against using character-class PCPs because users still find them annoying and some users will still fulfill requirements in predictable ways [3].

Even with the updated recommendations, character-class PCPs may remain ubiquitous, though they are largely unmeasured; the only previous study that explored PCPs on the web was from 2010 [20]. In our study, we measured the state of security and usability of PCPs present on the web by extracting them from websites we visited.

3 Study 1: password blocking

We measured whether popular websites prevent users from choosing the most common insecure passwords and found most of them insufficiently block users' choices. We selected common passwords to test based on two different strategies: blocking the 100,000 most frequently-used passwords found in password breaches (NCSC-HIBP-100k) and blocking passwords guessed early on by state-of-the-art cracking tools.

3.1 Method

3.1.1 We tested 120 of the top websites

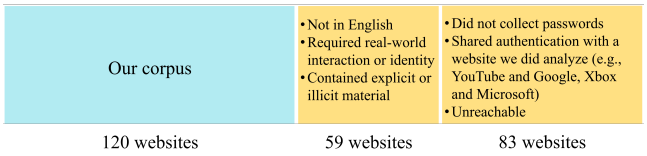


Figure 2: A breakdown of the 262 websites we attempted to study. We skipped 59 websites that did not fit our selection criteria. 83 websites either could not be analyzed or were already represented among our corpus of 120 websites.

In this study, we are concerned with password policies at the most popular English-language websites so our findings could be verified by all co-authors (who are all fluent in English). We focused on popular websites because previous research has shown that they generally have better security policies [20], which means that our results can be seen as an underestimate of conformity with best practices. Further, we wanted to hold these specific websites to account because they affect more users. Using an actively maintained ranked

[1] Modeling password strength with Shannon entropy is different from using guess numbers. Fig. 4 in § 6 illustrates this; character-class PCPs not only reject most weak passwords, but most strong passwords as well. See Appendix F for further discussion on the differences.

list provided by other researchers [26],[2] we tested the top 120 websites that were accessible to us. We skipped some websites for the reasons shown in Fig. 2; we reached our total of 120 websites after trying the top 262 listed entries.

Before the tests, we extracted the PCP on each website and encoded them in a regular expression (detailed in § 5.1). This allowed us to select PCP-compliant passwords for testing.

3.1.2 Testing common passwords leaked in breaches

We sampled 20 passwords from the NCSC-HIBP-100k list, which was ordered from most common to least common. We started by removing passwords that did not fit the website's PCP (with the aforementioned regular expressions) and sampling candidates to test at each website. In order to evenly represent the most frequently-leaked passwords along with the long tail of rest of the passwords in the list, we used a stratified sample based on powers-of-10 (1-10, 11-100, 101-1,000, 1,001-10,000, and 10,001-100,000). We randomly sampled candidates weighted by their position on the list ($\frac{1}{position}$), which gave us—in expectation—four passwords in each stratum. In order to ensure fair comparisons, websites with identical PCPs were tested with the same 20 passwords (e.g., all websites with a *1class8* PCP were tested with "`babygirl23`", "`lifeisgood`", etc.) We refer to these tested passwords as *leaked* passwords hereinafter.

Using the accounts we had set up, we attempted to change our password to each of the *leaked* passwords. If the change was successful, we logged out and logged back in with the new password to confirm, then noted that the password was accepted.

3.1.3 Testing common easy-to-guess passwords

In addition to restricting *leaked* passwords, websites should discourage users from selecting common passwords that are easily guessed (e.g., block "`Blink182`", which can be guessed in ~ 9 tries, or "`Hello123`", which can be guessed in ~ 316 tries). Here we tested the first 20 passwords that were guessed by state-of-the-art cracking tools at each website. We refer to these tested passwords as *easiest-guessed* passwords hereinafter.

We used Password Guessability Service (PGS)—offered by the Passwords Research Team at Carnegie Mellon University—to find these passwords to test [27]. PGS simulates a real attacker guessing passwords; it leverages multiple (5, at the time of our study) cracking tools to arrive at the user-provided plaintext password, returning the guess number (i.e., the number of guesses needed to find the password) as the password's strength rating. PGS also offers the `min_auto` configuration, which returns the minimum guess number for each password across all 5 tools. Previous research has found that the `min_auto` approach provides a conservative estimate for the performance of an unconstrained professional attacker [27]. Therefore, we referred to the `min_auto` guess number for all of the passwords in this study.

Since PGS requires its users to provide passwords in plaintext in order to receive results, we selected passwords to use from the Xato 10-million password dataset [28]. To the best of our knowledge, the dataset—which we will refer to as the Xato passwords hereinafter—represents the largest and most recent corpus of real-world passwords accessible to academic researchers, and has been widely used in previous work [3, 6, 16, 24]. We did search for newer password dumps to complement the Xato passwords, but found they were either available only on the dark web or offered in hashes rather than plaintext (to prevent large-scale cracking) [29, 30].

With all of the Xato passwords rated, we used the 20 passwords with the lowest guess numbers as our *easiest-guessed* passwords, and tested whether websites allowed them to be used. As with our testing of *leaked* passwords, we only selected passwords that fit the website's PCP. We excluded passwords that were already in the *leaked* passwords, and selected the password with the next-lowest guess number instead. Here, we also tested the same 20 passwords across websites with identical PCPs to ensure fairness (e.g., all websites with a *DigSym6* PCP—6+ characters with 1 digit or symbol—were tested with "`jordan23`", "`jessica1`", etc.). Every selected password could be guessed within $10^{4.9}$ guesses, well within the threshold of online guessing attacks.

3.2 Results

1. **Most websites do not block *leaked* or *easiest-guessed passwords* at all.** 71 / 120 websites accepted all 40 passwords we tested. By allowing both *leaked* and *easiest-guessed* passwords, these websites put their users at risk of password compromise and subsequent account hijackings. Additionally, accounts at other websites may be at risk for compromise too; users often practice poor security hygiene by reusing their passwords across the web, so this misconception that their password was not blocked and therefore suitable can have widespread insecurity.

   These 71 websites span different industries, including e-commerce (Amazon), social media (TikTok), entertainment (Netflix), and news (Wall Street Journal). Amazon, for instance, allowed us to change our password to "`123456`", the most common password on the web. TikTok—despite requiring users to choose a *3class8* password—allowed us to use "`p@ssw0rd`" (guessed by PGS in 7 tries, the fourth-most common *3class8* password) on our account.

2. **Additionally, several websites had insufficient blocking.** In addition to the 71 websites which accepted all 40 passwords, 19 sites accepted more than half of the *leaked* or *easiest-guessed passwords* tested. In some of these cases, this was likely due to insufficient blocklists. For example, IBM seemed to only block choices containing the word "password", which only blocked 1

[2] Available at `https://tranco-list.eu/list/VJ5N`. Generated on 29 July 2021.

("Password1") of the 40 passwords we tested on its site. Samsung only blocked number sequences (e.g., "123"), and Salesforce only blocked "password". While this may prevent users from using the most guessable passwords, the majority of the most common passwords still get accepted.

3. **10 websites seemed to be using a shorter *leaked* passwords blocklist.** We found 10 websites that blocked most of the tested *leaked* passwords from the higher-rank strata (e.g., 1-10, 11-100) but then allowed a majority of leaked passwords from the lower-rank strata (e.g., 10001-100000). This could indicate that these websites are using a truncated version of the NCSC-HIBP-100k list to check passwords, sacrificing security for usability. Spotify, for instance, blocked all *leaked* passwords up to the 101-1000 stratum but allowed all passwords beyond that point, which suggests that it only checks for the top 1000 *leaked* passwords. Our finding here is tentative, however, since we assumed websites were using the NCSC-HIBP-100k list. Table 5a in Appendix D shows the breakdown by stratum for the 10 websites.

4. **7 websites blocked *easiest-guessed* passwords, but not *leaked* passwords.** 7 websites disproportionately accepted more *leaked* passwords than *easiest-guessed* ones (Microsoft: 14 *leaked* accepted / 0 *easiest-guessed* accepted, LinkedIn: 14 / 0, WeTransfer: 19 / 4, Roblox: 18 / 6, Reddit: 16 / 4, Twitter: 12 / 2, and Indeed: 9 / 1).

   These 7 websites might have been using a minimum-strength requirement instead, since passwords they accepted generally had higher guess numbers than the passwords they rejected. If true, none of the websites set their minimum-strength requirement to prevent the threat of online guessing attacks (10^6 guesses). For example, Microsoft accepted one *leaked* password cracked with 251 guesses, and WeTransfer allowed one *leaked* password cracked with 32 guesses. Table 5b in Appendix D shows the minimum-strength cutoffs we found in our testing.

5. **Few websites prevented us from setting *leaked* and *easiest-guessed* passwords.** Only 15 websites—including Google, Adobe, Twitch, GitHub, and Grammarly—blocked all 40 passwords we tried. 7 more websites—including Apple, Canva, and VK—performed moderately well, allowing 5 or fewer tested passwords.

6. **Websites that allowed *leaked* and *easiest-guessed* passwords hold sensitive user information.** 38 of the 71 websites that allowed all 40 passwords store user payment information such as credit card or banking details, including Amazon, Netflix, GoDaddy, and Squarespace. 64 / 71 websites store PII about users, including Line, Intuit, Zoom, and MySpace. For each of the websites we analyzed, we checked whether it stored sensitive information using the test account we created earlier.

   While some websites could be low-risk, the majority of websites (70 / 120) we studied collect payment information and thus are potentially high-risk.

Appendix G presents a risk categorization of these 120 websites based on PII and payment information collection, along with the number of accepted *leaked* and *easiest-guessed* passwords.

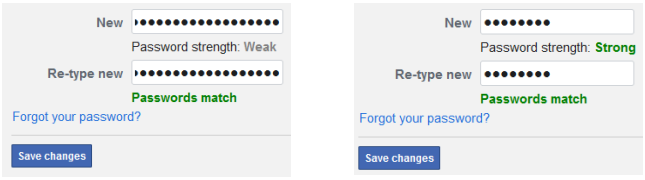
4 Study 2: strength meters

In 2014, de Carnavalet and Mannan investigated 11 password strength meters that were used in practice, and found most were estimating password complexity instead of guessability [5]. We wanted to know if there had been any changes; are meters at top sites now estimating guessability when a user chooses a password? To answer this, we reverse-engineered their patterns by testing different passwords.

4.1 Method

We considered all form elements on the password update page that updated in real-time to give feedback about the strength of the input on the password field. We then ran two tests on each of the strength meters to learn its patterns. First we investigated whether the meter was consistent in discouraging insecure choices; we tested the 100 *easiest-guessed* passwords from Xato that fit the website's PCP and noted the feedback received on each password. Next we tried to reverse-engineer the mechanics of the meter through boundary testing. We tested passwords with different lengths and number of character-classes, as well as passwords that were not compliant with the website's PCP. We selected passwords from Xato and also used passwords generated from password managers—Lastpass and 1Password—and noted movement patterns along the strength meter.

4.2 Results: most websites are not using strength meters to measure guessability



(a) "bkmmafwexucnvnsgppdk" (1 class, randomly generated) rated as Weak (1/3).

(b) "Passw0rd" (3 classes) rated as Strong (3/3).

Figure 3: Despite having a *1class6* PCP, Facebook's password strength meter is driven by adding more character-classes, and not password strength.

1. **Password strength meters are not widely used.** We found only 23 websites using password strength meters of any sort. Despite previous research touting the added

Password strength meters at websites with:	Our finding(s)	Implications on users	Prevalence
Minimum-length PCPs	• **Encourages complex passwords over passwords that are harder-to-guess.** Rates *easiest-guessed* passwords that have more character-classes higher than passwords with high guess numbers but fewer character-classes. • Discourages users' choices by nudging them toward fulfilling character-class PCPs.	• Password strength feedback does not reflect password guessability. • Possible usability issues similar to when fulfilling character-class PCPs.	6 / 18
Character-class PCPs	• **Encourages more complexity than required.** Meter tops out only if passwords include more character-classes than required by the PCP.	• Password strength feedback does not reflect password guessability. • Potential usability issues when a candidate password meets all stated requirements but does not fill the meter.	4 / 5

Table 2: Of the 23 websites that used password strength meters, 10 used those meters to encourage more complex passwords. 6 websites with minimum-length PCPs were actually using their meters as proxies for character-class PCPs.

security and usability benefits of using strength meters and robust open-source implementations like `zxcvbn`, most websites have not updated their password change procedures.

Regarding recommended strength meters, we found only two websites using `zxcvbn`; Dropbox (the organization behind the meter) and CPanel. The rest of the websites were using black box implementations that may not have been rigorously tested by the research community.

2. **10 / 23 websites misuse strength meters to measure complexity instead.** Rather than measuring password guessability, we found meters were actually being used as proxies for character-class PCPs. We break down our results by PCP here and in Table 2:

   6 / 18 websites with minimum-length PCPs use strength meters as character-class PCP nudges. Their strength meters would only increase if a password had more character-classes than the one entered prior, and not if it had a higher guess number. Despite Facebook's *1class6* PCP, its 3-point strength meter—shown in Fig. 3—considered all-lowercase passwords weak; "`zcdplgbtqldecfrzdqrw`" (randomly generated) was considered Weak (1/3) while "`Password1`" was considered Strong (3/3). The strength meter at Yelp (*1class6* PCP) unconditionally considered 16-character passwords strong, while requiring shorter passwords contain all four character-classes to be considered as such; "`123456789123456789`" (guess number ~ 631) was considered Great (4/4) while "`WzNGVE5uuWHd`" (randomly generated) was considered only Good (3/4). Since these meters measured complexity instead of estimating guessability, their readings were not reflective of how difficult it would be for an adversary to crack the password. Furthermore, users are nudged into creating complex passwords at these sites. The other 12 websites with minimum-length PCPs—including Google, Yahoo, and Twitch—had strength meters that more closely corresponded with password guessability; they rated all 100 passwords we tested as weak (<50% on their respective meters), and we did not find any insecure patterns when testing passwords with different character-classes and length.

   4 / 5 websites with character-class PCPs use their meters to encourage further complexity. They reserved the highest ratings on their meters for passwords that went beyond the required character-classes. Apple's strength meter, for instance, would only reach 100% if the password was 16-characters long and contained a symbol, despite its corresponding *3class8* PCP not requiring symbols. Aliexpress's 3-point meter only topped out if all 4 character-classes were included, despite a *2class6* PCP; "`jmDy&!py$Df&^tw*iBYy`" (randomly generated 3-class) was rated Middle (2/3), yet "`Abc123!@#`" (guess number ~ 53) evaluated to High (3/3). Since users may already be led by these sites to believe that compliance with character-class requirements would automatically yield strong passwords, they may find it frustrating when their password does not top up the strength meter. We only found one website—ScienceDirect—which did not encourage further complexity, only because its meter already filled up completely upon PCP-compliance.

3. **12 / 23 websites were inconsistent between meter feedback and password acceptance.** We then raised

the question: is the feedback from the meter on the user-side consistent with the ultimate decision by the website to accept or reject a user's chosen password on the server-side? Here, we used findings from our password blocking analysis—in which we had selected the 20 *easiest-guessed* Xato passwords that were compliant with a website's PCP and tested whether the website would accept them (§ 3)—and compared them with feedback given by the website's password strength meter. We now focused on feedback given by the website's strength meter right before submitting each password to the server. For each password, if feedback from the strength meter was negative (i.e., <50% of the scale), we coded user-side feedback as "unacceptable," otherwise, we coded the feedback as "acceptable."[3]

12 / 23 websites had varying levels of inconsistency. 5 websites rated all 20 passwords as "unacceptable," yet the server allowed all of them to be used; these websites rely solely on their strength meters, and do not perform additional checks before updating passwords. At CPanel, all 20 tested passwords were "unacceptable" (we found it was using `zxcvbn`), yet the server only rejected 13, which had all-letters or all-repeating-digits (e.g., rejected "`66666666`" but accepted "`12345`").

Only 11 / 23 websites were consistent between their strength meter feedback on the user-side and acceptance on the server-side. 8 of those sites—including W3C, Tumblr, and TechCrunch—rated all 20 passwords as "unacceptable," and all 20 were ultimately rejected. The other 3 sites were consistent in the opposite manner; they rated all 20 passwords "acceptable," and all 20 were ultimately accepted. Overall, these inconsistencies can lead to insecurity stemming from users unknowingly setting *easiest-guessed* passwords, as well as frustration when a user is told a password is good enough but is rejected.

Our key finding is that despite the usefulness of password strength meters being established in the research literature, adoption has remained low, and 10 / 23 of the sites that have them—6 of which have minimum-length PCPs—actually misuse them as proxies for character-class PCPs.

5 Study 3: composition policies

5.1 Method

5.1.1 We extracted the PCPs on 120 of the top sites

Using the aforementioned Tranco list (§ 3.1), we visited the top 120 websites that were accessible to us. At each website, we created an account and subsequently navigated to the password change page to reverse-engineer the website's PCP. We chose to use the password change page over the account creation page in order to avoid the need to repeatedly create new accounts and enter sign-up information (e.g., usernames, email addresses, names).

We noted the static creation rules that loaded on the form, then extracted dynamic rules by varying the password input with sample strings we had prepared in advance. We varied our sample strings to include strings with 1 class only (all lowercase letters, all digits, etc.) and strings with multiple classes (uppercase, lowercase, digits, and symbols). For symbols (i.e., special characters), we limited our permutations to the 33 ASCII characters that could be typed on a standard U.S. keyboard (shown below; note presence of the space character):

```
!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

We prioritized completeness in our method. For each PCP, we input and submitted non-compliant sample strings to make sure the website was enforcing its shown PCP. We also tested classes and characters that were not explicitly stated (e.g., for a hypothetical "`include at least one number`" character-class PCP we tested a sample string with only numbers to make sure there was no letter requirement, for a vague "`include a special character such as !#@(` " PCP we tested all 33 symbols and occasionally found websites that 1) counted other symbols towards the requirement, 2) allowed but did not count other symbols towards the requirement, or 3) disallowed other symbols entirely. The entire extraction task was done by hand and recorded in a spreadsheet (see Appendix C for a discussion of our attempts at an automated pipeline) by one of the co-authors and verified for correctness by a second co-author. After verification, we encoded the raw text of each website's PCP into a regular expression (which was also verified by at least two co-authors). The regular expressions were later used for other analyses in our study. We ended up with 73 different regular expressions (hence, 73 distinct PCPs among the 120 websites).

5.2 Results: character-class PCPs are still widely used

Character-class requirement	Websites (N=54)
Lowercase letters	31 (57.4%)
Uppercase letters	30 (55.6%)
Letters (case-insensitive)	19 (35.2%)
Digits	53 (98.1%)
Symbols (special characters)	37 (68.5%)

Table 3: Character-class requirements on the 54 websites with character-class PCPs. Nearly all require that passwords include a digit.

Our findings are as follows:

1. **Character-class PCPs are still widely used.** 54 websites (45%) still require users to include specific character classes in their password, despite recommendations

[3] Fortunately, we did not have to deal with any ambiguity between scale readings and labels on the meters we saw; all points below 50% had negative feedback, and all points 50% and above had neutral or positive feedback.

against these requirements. As found in previous studies, character-class PCPs impose a huge usability cost for a minimal security benefit [17, 19]. Table 3 shows the breakdown of the required character-classes. Almost all character-class PCPs require a digit, and symbols were the second-most popular requirement.

2. **Websites with character-class PCPs are more likely to allow the most common insecure passwords.** Cross-referencing our findings from § 3.2, we found that 38 of the 54 websites (70.4%) with character-class PCPs accepted all 40 of the *leaked* and *easiest-guessed* passwords we tried, compared with 33 of the remaining 66 websites (50%) with minimum-length PCPs. This may suggest that websites believe that complexity requirements are sufficient in getting users to create strong passwords, so they do not need to check passwords on a case-by-case basis.

3. **The most common minimum-length requirement is now 8 characters.** In 2010, Bonneau and Preibusch found that 52% of websites studied were using 6-character minimum length—followed by 4 characters (14%) and 5 characters (10%)—and that less than 5% of websites studied had an 8-character minimum length [20]. In our results over a decade later, we found that 66 / 120 websites studied (55%) have an 8-character minimum length, followed by 6 characters (35 / 120) and 5 characters (7 / 120). Perhaps this is a result of updated guidance from NIST in 2017, which now recommends an 8-character minimum length for passwords, up from its previous recommendation of 6 characters [1, 31].

4. **9 websites had inconsistencies between the PCP and text shown.** 1 website mentioned only a minimum-length requirement, but we were unable to save our password unless it contained a digit or a symbol. 2 other websites similarly failed to mention an additional character-class requirement in their text, which we uncovered through our testing. On the flip side, 4 websites did not enforce all of the character-class requirements mentioned. For example, Canva seemed to require us to include "`a mix of letters, numbers & symbols`" in our password, but we found that there were actually no character-class requirements. 1 website mentioned that whitespace characters were not allowed, but still accepted our password containing it. Lastly, 1 website with a *2class8* PCP had no text at all. We were only able to reverse-engineer its PCP after opening up development tools on our browser to view the server responses and making multiple attempts with different character-class combinations. Overall, these inconsistencies can lead to a confusing user experience.

Our key finding is that character-class PCPs are still being used on 45% of popular websites, burdening users while providing minimal security benefit. Even with the research against these complexity requirements, websites continue to force users to include extra characters like digits or symbols in their passwords, which some users may respond to in predictable ways. Furthermore, over 70% sites that continue to use character-class PCPs do not have any other password checks in place, allowing *leaked* and *easiest-guessed* passwords to be used.

We document several additional findings in Appendix E.

6 Study 4: PCP security and usability

In previous studies on PCPs, researchers typically conducted user studies by recruiting thousands of participants online (e.g., on Amazon Mechanical Turk) to perform password creation tasks on a testbed website. They would then analyze the passwords created for each PCP, such as measuring the complexity (entropy) of passwords created under each condition, the fraction of passwords guessed at a given guessing threshold, number of failed attempts, user sentiment, time taken to create a compliant password, and password recall rate [2, 12, 16]. These studies have influenced changes in password best practices over the past decade, particularly with the recommendation against character-class PCPs [1].

While it would certainly be useful to perform the same kind of password creation study for real-world PCPs, this was not feasible due to the large number of experimental conditions. As mentioned in § 5, we uncovered 73 unique PCPs among the 120 studied. For reference, in a previous user study, the authors recruited 5,099 participants who were assigned to 15 different PCPs; in order for us to replicate that power, we would have to recruit nearly 5 times as many participants [3]. These previous studies have also recognized the same limitations, and have kept the number of PCPs tested relatively small [2, 16, 17].

We therefore devised a different approach to measure the security and usability of PCPs studied. As we will show, our method has both advantages and limitations. Therefore our findings are tentative, and are ultimately intended to complement the findings from previous studies by providing insight into PCPs in practice.

6.1 Method

The fundamental insight of our method is to consider a PCP as a binary classifier, whose goal is to reject weak passwords and accept strong, hard-to-guess passwords. Here, we defined a password as weak if PGS could guess it in an online attack (within 10^6 guesses), and strong otherwise.

6.1.1 We assumed a corpus of passwords created without constraint

Users have different ways of generating passwords that are not influenced by a website's PCP [32]. Some examples include:

- Using a fixed password for all websites (password reuse)

- Using a password manager to automatically generate passwords
- Using a fixed heuristic (e.g., dictionary word + digit)

For our analysis we needed a sample of these "unconstrained passwords" to make unbiased comparisons of security and usability across PCPs. Our sample used here is the Xato 10-million passwords set (56% / 44% split between strong and weak passwords) [28]. Even though it did not meet our requirement of passwords generated without constraint—because users were already subject to the PCPs of the breached websites in this set—we still used it for analysis. We discuss the implications of using the Xato passwords later on.

6.1.2 We used sensitivity as a proxy for security

We assumed that whenever a user sets a new password at each of the 120 websites we studied, they initially generate one using an unconstrained strategy. If allowed by the PCP, the user will then confirm and set the password. Any PCP will allow some fraction of weak passwords through, however, which is why we measured the sensitivity of the PCP. We consider sensitivity—the percentage of weak passwords rejected—as a proxy for security. A website that simply allows any password to be used (i.e., no PCP), for example, would have 0% sensitivity.

One advantage of using sensitivity is that it is unaffected by outliers. Some generated passwords may have extremely high guess numbers and thus skew the average strength of passwords accepted by a PCP, for example. Since we used PGS to obtain guess numbers for the Xato passwords, we also benefited from accurate password strength ratings, as opposed to using entropy [25]. Our method to measure security has one disadvantage, however. Unlike in an intervention study, we don't know how users will react to any of the 120 PCPs, such as whether users go on to create strong passwords [12].

6.1.3 We used specificity as a proxy for usability

Some users—given their strategy for unconstrained password generation—will be frustrated by the PCP and be forced to pick a different strategy and password. While the usability cost would be justified if their password was actually weak, it would not be justified if it was strong. In the case of a password manager being incompatible with a PCP, they may be forced to pick a password manually, making it both weaker and less memorable (see § 2). Here, we used specificity to measure this usability cost. We used this measure as a proxy for usability of the PCP.

Specificity is an objective measure that complements other usability measures, like recall, user dropout, and time taken to enter a password. The main disadvantage to using specificity, however, is our inability to gauge user sentiment. That is, users may not necessarily feel frustrated by the PCP if their unconstrained password generation strategy is unsuccessful, especially if they have repeatedly encountered the same kind of PCP and have (predictable) adaptation strategies, or if their password manager accommodates them [33].

6.1.4 Limitations of using Xato passwords

Finally, we revisit the assumption about having a corpus of unconstrained passwords. Unfortunately, the Xato passwords set does not satisfy this requirement. While it is incredibly diverse—with weighted samples of over 1,000 password dumps collected over at least 5 years—most of the passwords were probably created by users reacting to some PCP [34]. One advantage to using the dataset, however, is that it doesn't contain passwords that required cracking [34]. This means that there is no bias towards weaker passwords.

Ultimately, using the Xato passwords in our security / usability evaluation means that we will overestimate usability (e.g., the segment created under the same PCP as the one being tested will have 100% usability) and underestimate security (e.g., the segment created under the same PCP as the one being tested will have 0% security). We reiterate that our findings in this section should be regarded as tentative; yet the strong limitations of PCPs that they reveal call into question the usefulness of PCPs and call for further research using different corpora and/or methods. For instance, a future user study could ask users to create passwords under no constraint (i.e., "`include at least 1 character`") and make that password set available for other researchers to use.

6.2 Results

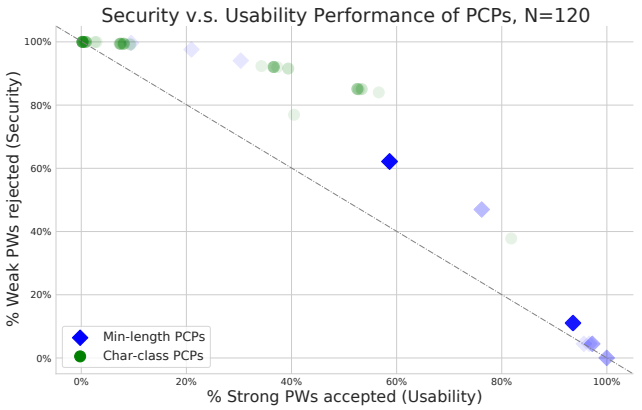


Figure 4: Scatter plot of security v.s. usability of PCPs for 120 websites. Each data point was plotted with 10% opacity, so more opaque areas reflect higher concentrations of PCPs with close scores.

Fig. 4 shows the scores of all websites we examined plotted along security and usability scores. Most of the 120 websites fall into one of three clusters: good security but poor usability (on the top left), good usability but poor security (bottom right), and average security and usability (in the middle of the graph). For comparison to a baseline, we also plotted a hypothetical PCP that rejects a random proportion α of passwords (and accepts $1-\alpha$ of passwords), the diagonal line represents that PCP's security and usability scores for $0 \leq \alpha \leq 1$. Our findings are as follows:

1. **No PCP simultaneously had more than 60% security and usability.** They either rejected too many strong passwords or accepted too many weak ones. Note that a hypothetical random PCP that blocks 50% of passwords has 50% security and usability simultaneously.

2. **PCPs fall on different parts of the security-usability spectrum.** Our results suggest a classic security-usability tension among PCPs. 69 / 120 websites we studied take opposite stances on the tradeoff; 33 have lenient policies (poor security cluster) and 36 have overly-stringent policies (poor usability cluster). Unsurprisingly, the PCPs within each of the 2 clusters are very similar to one another, with *1class6* being the majority PCP in the poor security cluster and *3class8* the majority in the poor usability cluster. We hypothesize that any PCP cannot be usable without allowing some weak passwords, and it cannot be secure without rejecting some strong password candidates.

   1class8 policies make up most of the PCPs with middling acceptance rates, rejecting only 62% of weak passwords and accepting only 58% of strong passwords. While considered to be a best practice, our results suggest that the PCP alone is insufficient in preventing users from choosing weak passwords; websites need to have additional safeguards—such as blocklists—to filter out the remaining 38%. This was not the case at nine *1class8* websites, including SoundCloud, Eventbrite, and Trello; they allowed all of the *leaked* and *easiest-guessed* passwords we tried, like "`1234qwer`","`1234567890`", and "`babygirl23`" (cross-referencing our findings from § 3).

3. **Most websites with insecure PCPs do not prevent insecure password choices.** 22 / 33 websites in the poor security cluster—including Amazon, Fox News, Etsy, and Dropbox (all with a *1class6* PCP)—do not block users from choosing passwords like "`abc123`" and "`qwerty`"—which we found with our password blocking analysis (§ 3)—and two more have insufficient blocking strategies (Slack and Yelp).

Our key finding is that PCPs are unsatisfactory in one or more ways. None of the 120 PCPs had more than 60% security and usability simultaneously. We hypothesize that there is no perfectly secure and usable PCP; all composition policies must make a tradeoff between user convenience (minimum-length PCPs) and strong passwords (character-class PCPs). Future studies should further investigate this hypothesis with different password corpora and methods. While websites with lenient PCPs can moderate the security gap with additional interventions like blocklists, we see this is not typically the case. A majority of these sites allow *leaked* and *easiest-guessed* passwords to be used.

7 Limitations

7.1 Limitations of analyzing the most popular websites

In these studies, we focused on the most popular websites. Since we did not additionally examine password policies of websites at the long tail (due to the work required to manually visit each website), we cannot be confident that our findings generalize to all websites. But note that previous research suggests that long-tail websites are likely to have even weaker security policies [20].

In Appendix B, we detail the access failures encountered at 142 websites in the ranked list we used. While future research can make an effort to study some of their password policies (like at government websites), we don't believe their exclusion here affects our overall finding: most top websites are not following best practices in their password policies. Moreover, 83 / 142 excluded websites did not collect passwords, shared authentication with a website we already analyzed, or were unreachable (e.g., DNS, measurement links).

7.2 Limitations in the PCP security / usability analysis

In the PCP security / usability analysis, we rated the strength of all 10 million Xato passwords using PGS under no policy, which served as our ground truth. As PGS conservatively simulates an adversary cracking passwords, it also offers to configure guess number calculations under a particular PCP, since the adversary—who knows the website's PCP—can constrain their search space to guess passwords more efficiently (the default option is no policy). Uploaded passwords that were compliant with the selected policy (17 options at the time of writing) would then be guessed with modified approaches using each of the cracking tools [27]. Since we did not select a policy to use, our results may lead to slight overreportings of both the fraction of strong passwords accepted and the fraction of weak passwords rejected for some of the PCPs. Obtaining more accurate ground truth measures would be challenging: PGS limits submissions to 30,000 passwords in order to ensure fair use of their free service, and their cracking tools can take a few weeks to complete.[4]

We did, however, further investigate the ramifications of using the no-policy guess numbers in our security / usability analysis, and found our main findings still hold true. We found that the *false positive risk*—the probability that a password rated strong was actually weak—was less than 4.56% at the 10^6 guesses threshold we used, for all 120 PCPs.[5] We randomly sampled 30,000 compliant passwords—weighted by their frequency—for each PCP and obtained their "PCP-aware" guess numbers from PGS in order to make the pairwise comparisons.

[4] The Passwords Research Team allowed us to submit all 10 million Xato passwords at once—for cracking under no policy—as a courtesy.

[5] Here we are concerned with the probability of a positive result being false [35]. This is different from the type 1 error rate (the false positive rate).

7.3 Limitations in scale

Our study required a significant amount of manual work to learn all of the password policies. For example, in our blocklist analysis alone, we attempted 4,800 password changes to determine whether websites allowed *leaked* and *easiest-guessed* passwords (∼200 hours of work). Since we manually visited each website to reverse-engineer their password policy, we were only able to test 120 of the top English-language websites. We hence did not try to draw statistically valid conclusions about differences between industry sectors (e.g. news vs. social media websites) because of the small number of websites. We leave those topics (e.g., how the rates of compliance with best practices might vary by rank, geographic location, or sector) as future research directions.

We initially attempted two automated approaches which we ultimately abandoned due to concerns with completeness and data quality. We include our experiences in Appendix C to hopefully serve as useful notes for those who want to extend our work.

8 Other related work

Some previous empirical works have partially looked at password policies in practice. Bonneau and Preibusch (2010) extracted the PCPs of 150 websites across 3 different site categories: identity providers, content providers, and e-commerce sites [20]. They found that identity providers were significantly more likely to have minimum-length requirements (>1 character password), character-class requirements, and basic dictionary checks whenever a user changes their password. Overall, they found poor adoption of industry standards for password implementations, such as using TLS, CAPTCHA, and rate-limiting password guesses.

de Carnavalet et al. (2015) studied the password strength meters used at 11 popular websites [5]. They extracted or reverse-engineered the meter implementation at each site to local scripts and ran large-scale automated tests to get strength readings of known passwords, running a total of 53 million tests. They found most meters were only measuring password complexity, with only one implementation—`zxcvbn`—going beyond to penalize dictionary words. They also found that among the password strength meters in use on the web, most of them were inaccurate and inconsistent; passwords rated weak were often rated strong at other websites, and vice versa.

We built on the work done in both studies to deliver new additional insights. For password strength meters, we found websites with minimum-length PCPs that were using their strength meters as character-class nudges (§ 4). We also focused on investigating consistency between meter feedback at the client and password acceptance at the server by attempting to set the 20 *easiest-guessed* passwords we tested, and found more than half of websites were inconsistent. For PCPs, we resurveyed the landscape over a decade later, and found changes in the types of requirements used (§ 5). We also developed a new method to measure the security and usability of PCPs, and tentatively found none of them had decent security and usability simultaneously (§ 6).

9 Conclusion

Even with the gains in user authentication methods over the past two decades, passwords remain essential for online access, and replacing them in the near future seems improbable [36]. For these reasons, online services—especially the websites in which we found flaws—need to focus on password security and usability. Websites with insufficient blocklisting strategies, an outdated character-class PCP, or a misconfigured password strength meter should review the best practices summarized in Table 1 and make adjustments to their password policies. We further encourage them to review the research behind the guidelines in order to avoid misconfigured interventions that are inconsistent with one another (e.g., § 4).

We also suggest future research that directly engages with system administrators, in order to understand their mindset on password security. Researchers may then be able to uncover the reasons for the disconnect between industry and the academic community, and take steps towards reconciling the disparity. Some hypotheses include:

- Password policy is security theater: measures such as character-class PCPs, even if ineffective, may give users a false sense of security, and websites use them for this reason.
- Websites have shifted their attention to adopting other authentication technologies, such as multi-factor authentication (MFA), and believe that it is unnecessary to strengthen their password policies. (Note that there are severe weaknesses in SMS-based MFA, so this view might be overoptimistic [37, 38]).
- Websites need to pass security audits, and the firms who do these audits, such as Deloitte, recommend or mandate outdated practices.
- Websites face some other practical constraint that the academic community does not know about.

We have made our dataset available for other researchers at: `https://passwordpolicies.cs.princeton.edu/`.

Acknowledgements

We are grateful to Ryan Amos, Ben Kaiser, Malte Möser, and our anonymous reviewers for their helpful feedback on our writeup. We thank Arunesh Mathur and Prateek Mittal for discussions on research questions at the beginning of the study. We are also grateful to Ross Anderson, Richard Clayton, and other participants at the Cambridge security seminar for their valuable feedback.

References

[1] Paul A. Grassi et al. *NIST Special Publication 800-63B Digital Identity Guidelines. Authentication and Lifecycle Management*. June 22, 2017. DOI: 10.6028/NIST.SP.800-63b.

[2] Patrick Gage Kelley et al. "Guess Again (and Again and Again): Measuring Password Strength by Simulating Password-Cracking Algorithms". In: *Proceedings of the 33rd IEEE Symposium on Security & Privacy (S&P)*. May 2012. DOI: 10.1109/SP.2012.38.

[3] Joshua Tan et al. "Practical Recommendations for Stronger, More Usable Passwords Combining Minimum-Strength, Minimum-Length, and Blocklist Requirements". In: *Proceedings of the 27th ACM SIGSAC Conference on Computer and Communications Security (CCS)*. Oct. 2020. DOI: 10.1145/3372297.3417882.

[4] Dan U. *Passwords, passwords everywhere*. National Cyber Security Centre. Apr. 21, 2019. URL: https://www.ncsc.gov.uk/blog-post/passwords-passwords-everywhere (visited on 12/21/2021).

[5] Xavier De Carné de Carnavalet and Mohammad Mannan. "From Very Weak to Very Strong: Analyzing Password-Strength Meters". In: *Proceedings of the 21st Network and Distributed System Security Symposium (NDSS)*. Feb. 2014. DOI: 10.14722/ndss.2014.23268.

[6] Blase Ur et al. "Design and Evaluation of a Data-Driven Password Meter". In: *Proceedings of the 2017 ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*. Apr. 2017. DOI: 10.1145/3025453.3026050.

[7] Richard Shay et al. "A Spoonful of Sugar? The Impact of Guidance and Feedback on Password-Creation Behavior". In: *Proceedings of the 2015 ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*. Apr. 2015. DOI: 10.1145/2702123.2702586.

[8] Joseph Bonneau. "The Science of Guessing: Analyzing an Anonymized Corpus of 70 Million Passwords". In: *Proceedings of the 33rd IEEE Symposium on Security & Privacy (S&P)*. May 2012. DOI: 10.1109/SP.2012.49.

[9] Matteo Dell'Amico, Pietro Michiardi, and Yves Roudier. "Password Strength: An Empirical Analysis". In: *2010 Proceedings IEEE INFOCOM*. Mar. 2010. DOI: 10.1109/INFCOM.2010.5461951.

[10] William Melicher et al. "Fast, Lean, and Accurate: Modeling Password Guessability Using Neural Networks". In: *Proceedings of the 25th USENIX Security Symposium (USENIX Security)*. Aug. 2016. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/melicher.

[11] Dinei Florêncio, Cormac Herley, and Paul C. van Oorschot. "An Administrator's Guide to Internet Password Research". In: *Proceedings of the 28th Large Installation System Administration Conference (LISA14)*. Nov. 2014. URL: https://www.usenix.org/conference/lisa14/conference-program/presentation/florencio.

[12] Saranga Komanduri et al. "Of Passwords and People: Measuring the Effect of Password-Composition Policies". In: *Proceedings of the 2011 ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*. Apr. 2011. DOI: 10.1145/1978942.1979321.

[13] Verizon DBIR Team. *2020 Data Breach Investigations Report*. May 19, 2020. URL: https://www.verizon.com/business/resources/reports/dbir/2020/ (visited on 03/22/2022).

[14] ESET. *ESET Threat Report T3 2021*. Feb. 9, 2022. URL: https://www.welivesecurity.com/wp-content/uploads/2022/02/eset_threat_report_t32021.pdf (visited on 03/22/2022).

[15] Steven Furnell. *Stop blaming people for choosing bad passwords – it's time websites did more to help*. The Conversation. Jan. 3, 2022. URL: https://theconversation.com/stop-blaming-people-for-choosing-bad-passwords-its-time-websites-did-more-to-help-172257 (visited on 01/26/2022).

[16] Hana Habib et al. "Password Creation in the Presence of Blacklists". In: *Proceedings of the 2017 Workshop on Usable Security (USEC)*. Feb. 2017. DOI: 10.14722/usec.2017.23043.

[17] Richard Shay et al. "Can Long Passwords Be Secure and Usable?" In: *Proceedings of the 2014 ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*. Apr. 2014. DOI: 10.1145/2556288.2557377.

[18] Richard Shay et al. "Encountering Stronger Password Requirements: User Attitudes and Behaviors". In: *Proceedings of the 6th Symposium On Usable Privacy and Security (SOUPS)*. July 2010. DOI: 10.1145/1837110.1837113.

[19] Blase Ur et al. ""I Added '!' at the End to Make It Secure": Observing Password Creation in the Lab". In: *Proceedings of the 11th Symposium On Usable Privacy and Security (SOUPS)*. July 2015. URL: https://www.usenix.org/conference/soups2015/proceedings/presentation/ur.

[20] Joseph Bonneau and Sören Preibusch. "The password thicket: technical and market failures in human authentication on the web". In: *The Ninth Workshop on the Economics of Information Security*. June 2010. URL: https://econinfosec.org/archive/weis2010/papers/session3/weis2010_bonneau.pdf.

[21] Matt Weir et al. "Password Cracking Using Probabilistic Context-Free Grammars". In: *Proceedings of the 2009 30th IEEE Symposium on Security and Privacy*. May 2009. DOI: 10.1109/SP.2009.8.

[22] Kurt Thomas et al. "Protecting accounts from credential stuffing with password breach alerting". In: *Proceedings of the 28th USENIX Security Symposium (USENIX Security)*. Aug. 2019. URL: https://www.usenix.org/conference/usenixsecurity19/presentation/thomas.

[23] Bijeeta Pal et al. "Might I Get Pwned: A Second Generation Compromised Credential Checking Service". In: *Proceedings of the 31st USENIX Security Symposium (USENIX Security)*. Aug. 2022. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/pal.

[24] Daniel Lowe Wheeler. "zxcvbn: Low-Budget Password Strength Estimation". In: *Proceedings of the 25th USENIX Security Symposium (USENIX Security)*. Aug. 2016. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/wheeler.

[25] Matt Weir et al. "Testing Metrics for Password Creation Policies by Attacking Large Sets of Revealed Passwords". In: *Proceedings of the 17th ACM SIGSAC Conference on Computer and Communications Security (CCS)*. Oct. 2010. DOI: 10.1145/1866307.1866327.

[26] Victor Le Pochat et al. "Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation". In: *Proceedings of the 26th Network and Distributed System Security Symposium (NDSS)*. Feb. 2019. DOI: 10.14722/ndss.2019.23386.

[27] Blase Ur et al. "Measuring Real-World Accuracies and Biases in Modeling Password Guessability". In: *Proceedings of the 24th USENIX Security Symposium (USENIX Security)*. Aug. 2015. URL: https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/ur.

[28] Mark Burnett. *Today I Am Releasing Ten Million Passwords*. Feb. 9, 2015. URL: https://xato.net/today-i-am-releasing-ten-million-passwords-b6278bbe7495 (visited on 01/06/2022).

[29] Troy Hunt. *Introducing 306 Million Freely Downloadable Pwned Passwords*. Aug. 3, 2017. URL: https://www.troyhunt.com/introducing-306-million-freely-downloadable-pwned-passwords/ (visited on 12/22/2021).

[30] Troy Hunt. *The 773 Million Record "Collection #1" Data Breach*. Jan. 17, 2019. URL: https://www.troyhunt.com/the-773-million-record-collection-1-data-reach/ (visited on 12/22/2021).

[31] William Burr et al. *NIST Special Publication 800-63-2 Electronic Authentication Guideline*. Aug. 2013. DOI: 10.6028/NIST.SP.800-63-2.

[32] Blase Ur et al. "Do Users' Perceptions of Password Security Match Reality?" In: *Proceedings of the 2016 ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*. May 2016. DOI: 10.1145/2858036.2858546.

[33] Apple. *Password Rules Validation Tool*. URL: https://developer.apple.com/password-rules/ (visited on 02/14/2022).

[34] Mark Burnett. *Ten Million Passwords FAQ*. Feb. 10, 2015. URL: https://xato.net/ten-million-passwords-faq-3b2752ed3b4c (visited on 01/06/2022).

[35] David Colquhoun. "The reproducibility of research and the misinterpretation of p-values". In: *Royal Society Open Science* 4 (12 Dec. 2017). DOI: 10.1098/rsos.171085.

[36] Joseph Bonneau et al. "The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes". In: *Proceedings of the 33rd IEEE Symposium on Security & Privacy (S&P)*. May 2012. DOI: 10.1109/SP.2012.44.

[37] Kevin Lee et al. "An Empirical Study of Wireless Carrier Authentication for SIM Swaps". In: *Proceedings of the 16th Symposium On Usable Privacy and Security (SOUPS)*. Aug. 2020. URL: https://www.usenix.org/conference/soups2020/presentation/lee.

[38] Kevin Lee and Arvind Narayanan. "Security and Privacy Risks of Number Recycling at Mobile Carriers in the United States". In: *Proceedings of the 2021 APWG Symposium on Electronic Crime Research (eCrime)*. Dec. 2021. DOI: 10.1109/eCrime54498.2021.9738792.

A Visualization of best practices

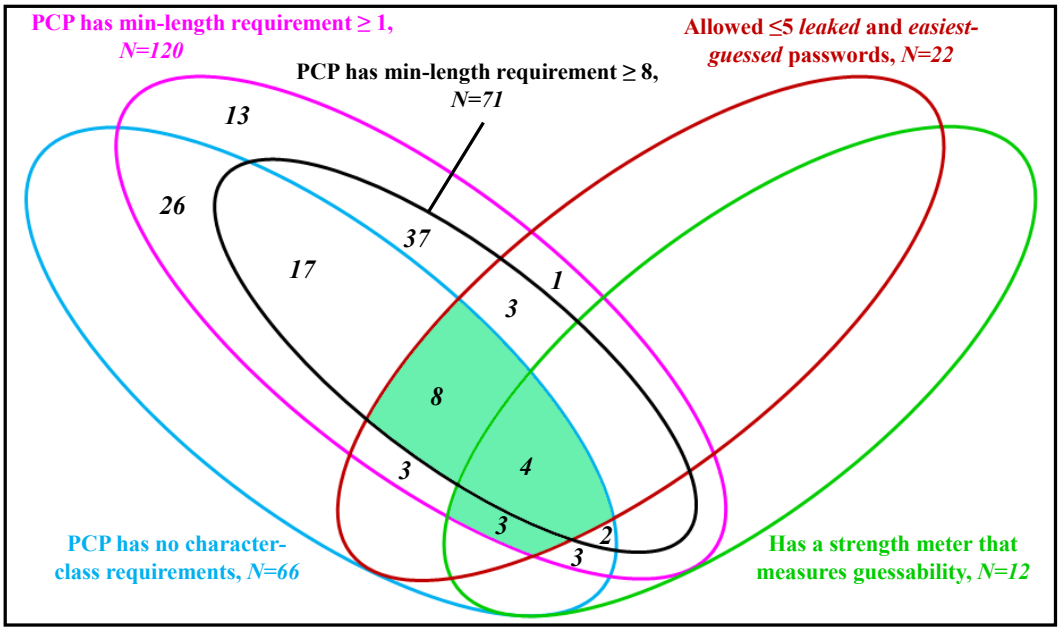


Figure 5: Websites following best practices are in the shaded green area. Unlabeled areas contain 0 websites.

Fig. 5 shows the breakdown of websites we considered to be following best practices. We considered a website to be following best practices if it allowed 5 or fewer of the 40 most common leaked passwords and easiest-to-guess passwords we tried, required passwords be no shorter than 8 characters, and did not impose any character-class requirements. We also considered websites with a shorter minimum-length requirement as following best practices if they satisfied the other two recommendations and further employed an accurate password strength meter to guide users to choosing strong passwords.

B Access failure details

Reason	Websites (N=142)
Inaccessible	69
No registration page	26
No passwords for auth	3
Government website	2
University website	4
Purchase required	7
Never received registration SMS	1
Non-U.S. phone number required	1
Site unreachable from browser	25
Explicit material	6
Non-English	38
Shared reg page w/ already-visited site	29

Table 4: Breakdown of the websites we skipped in our study.

We tried visiting the top 262 websites on the Tranco list in order to obtain the 120 websites for our study. Table 4 lists the reasons we skipped the other 142 websites.

C Lessons from our attempts at automation

Some readers may wonder why we pursued manual data sourcing methods in this study instead of an automated approach, since doing so may have enabled us to scale up the number of websites tested. As a matter of fact, we initially attempted two automated approaches which we ultimately abandoned due to concerns with completeness and data quality. We include our experiences in this writeup to hopefully serve as useful notes for those who want to extend our work.

We first tried building and using a Selenium-based web crawler to automatically extract PCPs from websites. Our crawler consisted of scripts tasked with parsing and navigating the sites of given domains to find the registration form and the PCP on the form. We leveraged search engine keyword searches to find registration pages (e.g., “join”, “create”, “signup”), as well as pattern detection of HTML tags and keywords to find and extract the PCP. However, we soon found that it was practically infeasible to develop any general solution; the unstandardized registration flows across websites required us to constantly add code to handle an extremely wide range of UI designs.

Our second approach utilizing MTurk was more successful,

	Fraction of accepted *leaked* passwords by stratum				
	1-10	11-100	101-1,000	1,001-10,000	10,001-100,000
bit.ly	0/2	1/3	1/4	1/6	3/5
chase.com	0/1	1/3	1/4	5/7	4/5
espn.com	0/2	1/3	2/4	6/6	5/5
facebook.com	0/2	1/3	0/4	3/6	4/5
instagram.com	0/2	1/3	0/4	3/6	4/5
slack.com	0/2	1/3	1/4	6/6	5/5
spotify.com	0/2	2/3	1/4	4/6	5/5
surveymonkey.com	0/2	2/3	1/4	4/6	4/5
tripadvisor.com	0/2	0/3	0/4	6/6	5/5
yelp.com	0/2	1/3	4/4	5/6	4/5

(a) Looking at accepted *leaked* passwords by stratum, we hypothesized 10 websites were using shorter verisons of the NCSC-HIBP-100k list.

	Hypothesized minimum-strength threshold
indeed.com	10^3
linkedin.com	$10^{2.3}$
microsoft.com	$10^{2.4}$
roblox.com	10^1
reddit.com	10^1
twitter.com	$10^{3.4}$
wetransfer.com	$10^{1.5}$

(b) Minimum-strength thresholds we hypothesized were being used at seven websites. For reference, the threat of online guessing attacks ends at 10^6 guesses.

Table 5: We found 10 websites that seemed to be blocking passwords based on a shorter common passwords list, and found 7 websites that seemed to be blocking passwords that did not meet a minimum-strength requirement.

but still produced data of dubious quality. We developed and published two separate MTurk Tasks to workers on the marketplace: one to identify registration pages from a given domain, and a second to extract the PCP from a given registration page. For each Task, the Worker—our hired user—was given the domain or registration page, and given a form to input information found such as the minimum-length requirement and character-class requirements. We also included quality assurance questions on the forms to confirm that the Worker had understood the given task and was paying attention. Despite the additional quality assurance measures, we found widespread inconsistency in the clarity of information collected across websites and even at the same website (we made sure to create two Tasks for each website). We concluded that our assurances were not rigorous enough, and that we had also underestimated the difficulty of educating Workers about extracting PCPs.

D Password blocking trends from § 3

Table 5 shows two trends we found in our password blocking study (§ 3). In Table 5a, we hypothesized 10 websites were using shorter versions of the most common passwords list we used. In Table 5b, we hypothesized 7 websites had a minimum-strength requirement.

E Additional findings from § 5

1. **Symbol definitions varied among the 37 websites requiring them.** 13 websites counted all 33 symbols we used towards their requirement, and half of the websites counted all but one symbol. The remaining websites below the median counted far fewer symbols, however, including 1 website that counted only `#`, `$`, `&`, and `@` (4 symbols), and 2 websites that counted only 9 and 10 symbols, respectively. The most commonly excluded symbol was the space character, which counted at only 15 / 37 websites, followed by the `'`, `"`, and `` ` `` characters, each counted as symbols at 28 / 37 websites.

   13 websites placed even more restrictions on certain symbols by outright disallowing them in passwords, including the 2 websites that counted only 9 and 10 symbols; any symbol that did not count was not allowed to be in the password.

2. ***1class8* is the most common PCP.** 24 / 120 websites (20%) were using this PCP, followed by *1class6* (22 / 120). *3class8* is the most common character-class PCP (and third-most popular overall), we found it on 17 websites, followed by *4class8* and *DigSym6*, each found being used on 10 websites.

3. **Some websites were using maximum-length requirements that are too short.** 17 websites had a maximum-length requirement below 64 characters—the baseline recommended by NIST—including 1 website with a 14-character maximum length, 3 with a 15-character maximum, and 4 with a 20-character maximum [1]. Setting too short of a maximum length hurts security by preventing users from choosing long passwords that are hard-to-guess.

F Additional background

F.1 Password security is better modeled through adversarial guessability

Password strength has traditionally been measured using Shannon entropy, a function of the counts of lower- and uppercase letters, digits, and symbols (LUDS). While previously recommended by NIST, entropy—also commonly referred to as complexity—turned out to be a poor proxy for password security [31]. Researchers soon found mismatches between password entropy scores and time needed for attackers to

crack a password (or a set of passwords) [25]. The information security community has since favored using guessability as a measure of password security [8, 9].

Guessability more closely resembles the only practically important sense of password strength: the actual number of guesses an adversary would require to correctly guess the password. Unlike Shannon entropy, guess number metrics can factor in contextual information such as common passwords, human predictability and composition rules presented at password creation [25]. However, the attack method and configuration matters: many previous studies—facing time and resource constraints—have only been able to model specific attackers by using only one attack method with limited training data. A necessary drawback to the guessability approach is its inherent subjectivity. Whereas entropy is an objective measure, there is no objective guess number for any password; adversarial guessing is a strategic problem and different strategies will produce different results over the same password set input. For this reason, comparisons between studies using different guessing algorithms can be difficult at best, and moot at worst.

In an effort to harmonize future studies, in 2015, the Passwords Research Team at Carnegie Mellon University released Password Guessability Service (PGS)—a free service that rates the strength of submitted passwords [27]. PGS simulates a real attacker guessing passwords; it leverages multiple (5 at the time of our study) cracking tools to arrive at the user-provided plaintext password. Using each tool, PGS calculates the guessability (i.e., the guess number) as the password's strength rating. PGS also offers the `min_auto` configuration, which returns the minimum guess number for each password across all 5 tools. Previous research has found that the `min_auto` approach provides a conservative estimate for the performance of an unconstrained professional attacker [27].

Table 6: PCP, blocklist results, strength meter, and security / usability ratings.

Website	Rank	Stores payment information	Stores PII	PCP	Allowed *leaked*	Allowed *easiest-guessed*	Strength meter	Percentage weak passwords rejected (security proxy)	Percentage strong passwords accepted (usability proxy)
google.com	1	•	•	*1class8*	0	0	Models guessability	62%	59%
netflix.com	2	•	•	*1class6*	20	20		11%	94%
facebook.com	4	•	•	*1class6*	8	2	Models complexity	11%	94%
microsoft.com	5	•	•	*2class8*	14	0		92%	39%
twitter.com	6		•	*1class8*	12	2		62%	59%
instagram.com	7	•	•	*1class6*	8	3		11%	94%
linkedin.com	9	•	•	*1class8*	14	0		62%	59%
apple.com	11	•	•	*3class8*	4	0	Models complexity	99%	7%
wikipedia.org	12			*1class8*	5	1		62%	59%
amazon.com	16	•	•	*1class6*	20	20		11%	94%
yahoo.com	17		•	*1class7*	0	0	Models guessability	47%	76%
pinterest.com	21		•	*1class6*	4	1		11%	94%
adobe.com	22	•	•	*3class8*	0	0		99%	8%
vimeo.com	24	•	•	*2class8*	18	15		100%	1%
wordpress.com	27	•	•	*1class6*	3	1		11%	94%
reddit.com	31			*1class8*	16	4		62%	59%
zoom.us	33		•	*3class8*	20	20		99%	7%
github.com	34	•	•	*1class15* or *2class8*	0	0		92%	36%
amazonaws.com	36	•	•	*3class8*	20	20		99%	8%
bit.ly	37	•		*1class6*	6	3		11%	94%
tumblr.com	43		•	*1class8*	0	0	Models guessability	62%	59%
vk.com	48	•	•	*1class6*	1	1		11%	94%
nytimes.com	49	•	•	*1class6*	20	20		11%	94%
flickr.com	51	•	•	*1class12*	20	20		100%	9%
dropbox.com	53	•	•	*1class6*	20	20	Models guessability	11%	94%
soundcloud.com	56	•	•	*1class8*	20	20		62%	59%
spotify.com	59	•	•	*1class8*	12	6		62%	59%
myshopify.com	60	•	•	*1class5*	20	20	Models guessability	4%	97%
cnn.com	65		•	*4class8*	20	20		100%	0%
forbes.com	66	•	•	*4class8*	20	20		100%	0%
ebay.com	68	•	•	*DigSym6*	11	9		85%	53%
theguardian.com	69	•	•	*1class8*	0	0		62%	59%
w3.org	70		•	*1class8*	0	0	Models complexity	62%	59%
paypal.com	72	•	•	*DigSym8*	15	5		77%	40%
twitch.tv	73		•	*1class8*	0	0	Models guessability	62%	59%
sourceforge.net	74		•	*1class10*	0	0		98%	21%
cloudflare.com	75	•		*2class8*	20	20		100%	1%
archive.org	76		•	*1class3*	20	20		0%	100%
imdb.com	77		•	*1class8*	20	20		62%	59%
bbc.co.uk	89		•	*2class8*	17	14		92%	37%
issuu.com	91	•	•	*1class4*	20	20		0%	100%
weebly.com	92	•	•	*1class8*	0	0		62%	59%
aliexpress.com	95	•	•	*2class6*	20	20	Models complexity	84%	57%
washingtonpost.com	96	•	•	*1class8*	20	20		62%	59%

Continued on next page

Table 6: *(Continued)* PCP, blocklist results, strength meter, and security / usability ratings.

Website	Rank	Stores payment information	Stores PII	PCP	Allowed *leaked*	Allowed *easiest-guessed*	Strength meter	Percentage weak passwords rejected (security proxy)	Percentage strong passwords accepted (usability proxy)
stackoverflow.com	98		•	*2class8*	20	20		92%	37%
etsy.com	99	•	•	*1class6*	20	20	Models complexity	11%	94%
reuters.com	103			*4class8*	20	20		100%	0%
tinyurl.com	106	•		*1class6*	20	20		11%	94%
tiktok.com	108		•	*3class8*	20	20		100%	1%
wsj.com	109		•	*2class5*	20	20		85%	53%
wix.com	113	•	•	*1class6*	20	20	Models complexity	11%	94%
bloomberg.com	114	•	•	*1class8*	2	1		62%	59%
sciencedirect.com	118		•	*4class8*	20	20	Models complexity	100%	0%
slideshare.net	120		•	*1class5*	20	20		4%	97%
imgur.com	121		•	*DigSym6*	20	20		85%	53%
oracle.com	122		•	*4class8*	20	20		100%	0%
opera.com	123		•	*1class8*	0	0		62%	59%
booking.com	125	•	•	*3class10*	20	20		100%	3%
indeed.com	126		•	*1class8*	9	1		62%	59%
businessinsider.com	127	•	•	*3class8*	20	20		99%	7%
canva.com	132	•	•	*1class8*	2	0	Models guessability	62%	59%
godaddy.com	135	•	•	*1class9*	20	20	Models guessability	94%	30%
cnet.com	140		•	*DigSym6*	20	20		100%	1%
ibm.com	143	•	•	*3class8*	19	20		99%	7%
researchgate.net	144		•	*1class6*	20	20	Models complexity	38%	82%
digicert.com	145	•		*3class8*	20	20		100%	3%
dailymail.co.uk	148		•	*1class5*	20	20		5%	96%
slack.com	150	•	•	*1class6*	13	4		11%	94%
fandom.com	154			*1class1*	20	20		0%	100%
nature.com	157		•	*2class8*	20	20		92%	37%
force.com	159	•	•	*2class8*	19	18		92%	37%
cnbc.com	160		•	*3class8*	20	20		100%	0%
usatoday.com	161	•	•	*1class5*	20	20	Models complexity	5%	97%
chase.com	163	•	•	*2class8*	11	9		92%	34%
walmart.com	164	•	•	*3class8*	20	20		99%	7%
hp.com	166	•	•	*3class8*	20	20		99%	8%
surveymonkey.com	168	•	•	*1class8*	11	2		62%	59%
aol.com	170		•	*1class7*	0	0	Models guessability	47%	76%
yelp.com	171		•	*1class6*	14	6	Models complexity	11%	94%
eventbrite.com	173	•	•	*1class8*	20	20	Models guessability	62%	59%
telegraph.co.uk	174	•	•	*1class8*	20	20		62%	59%
opendns.com	176		•	*4class8*	20	20		100%	0%
cpanel.net	177	•	•	*1class4*	7	7	Models guessability	0%	100%
springer.com	186			*DigSym6*	20	20		85%	53%
time.com	187	•	•	*3class8*	20	20		100%	0%
npr.org	189		•	*1class5*	20	20		4%	97%
ted.com	190	•	•	*1class8*	20	20		62%	59%
samsung.com	191	•	•	*3class8*	19	14		99%	8%
myspace.com	194		•	*2class8*	20	20		92%	39%

Continued on next page

Table 6: *(Continued)* PCP, blocklist results, strength meter, and security / usability ratings.

Website	Rank	Stores payment information	Stores PII	PCP	Allowed *leaked*	Allowed *easiest-guessed*	Strength meter	Percentage weak passwords rejected (security proxy)	Percentage strong passwords accepted (usability proxy)
dailymotion.com	196		•	*3class8*	20	20		100%	1%
themeforest.net	198	•	•	*1class8*	0	0		62%	59%
huffingtonpost.com	199		•	*3class8*	20	20		99%	8%
wired.com	200	•	•	*1class6*	20	20		11%	94%
mailchimp.com	201	•	•	*4class8*	20	20		100%	0%
espn.com	202		•	*DigSym6*	14	10	Models complexity	85%	53%
addthis.com	204			*1class6*	20	20		11%	94%
techcrunch.com	205		•	*1class7*	0	0	Models guessability	47%	76%
scribd.com	208	•	•	*1class8*	20	20		62%	59%
zillow.com	211		•	*4class8*	20	20		100%	0%
goodreads.com	212		•	*1class6*	20	20		11%	94%
unsplash.com	213		•	*1class6*	20	20		11%	94%
indiatimes.com	214		•	*DigSym6*	20	20		100%	1%
trello.com	219	•	•	*1class8*	20	20		62%	59%
grammarly.com	220	•	•	*1class8*	0	0		62%	59%
tripadvisor.com	221	•	•	*1class6*	11	3		11%	94%
freepik.com	222	•	•	*DigSym6*	0	2		100%	0%
independent.co.uk	225	•	•	*DigSym6*	20	20		99%	9%
roblox.com	226	•	•	*1class8*	18	6		62%	59%
squarespace.com	230	•	•	*1class6*	20	20		11%	94%
foxnews.com	232		•	*1class6*	20	20		11%	94%
zendesk.com	237	•	•	*1class5*	20	20		4%	97%
latimes.com	239	•	•	*DigSym6*	20	20		85%	53%
line.me	245		•	*DigSym6*	20	20		85%	52%
shutterstock.com	246	•	•	*1class8*	20	20		62%	59%
livejournal.com	247		•	*DigSym6*	19	10		99%	9%
wetransfer.com	248	•	•	*3class8*	19	4		99%	8%
intuit.com	250	•	•	*4class8*	20	20		100%	0%
intel.com	254		•	*3class8*	20	20		100%	1%
stackexchange.com	256		•	*2class8*	20	20		92%	37%
w3schools.com	262	•	•	*4class8*	20	20		100%	0%