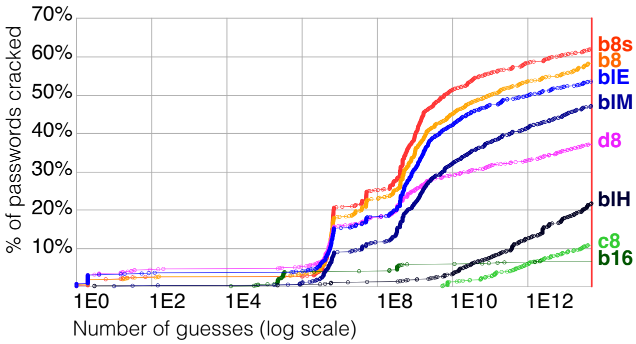

Figure 2. The number of passwords cracked vs. the number of guesses, per condition, for experiment P4. This experiment uses the Weir calculator and trains on a variety of publicly available data.

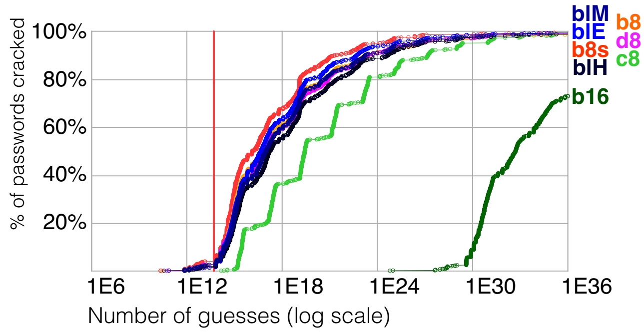

Figure 3. The number of passwords cracked vs. the number of guesses, using the BFM calculator trained on both our data and public data (B2). The red vertical line at 50 trillion guesses facilitates comparison with the Weir experiments. We stopped the Weir calculator at this point (as described in Section IV-A3), but because the BFM algorithm is so much less efficient, we ran it for many more guesses in order to collect useful data.

passwords from each of our eight conditions. We test on 500 other passwords from those conditions, with two-fold cross-validation for a total of 1000 test passwords. The results from these experiments are shown in Figures 1 and 2.

As these figures suggest, which password-composition policy is best at resisting guessing attacks depends on how many guesses an attacker will make. At one million and one billion guesses in both experiments, significantly fewer blacklistHard and comprehensive8 passwords were guessed than in any other condition.[3] At one billion guesses in experiment E, 1.4, 2.9, 9.5, and 40.3% of passwords were cracked in comprehensive8, blacklistHard, basic16, and basic8, respectively.

As the number of guesses increases, basic16 begins to outperform the other conditions. At one trillion guesses, significantly fewer basic16 passwords were cracked than comprehensive8 passwords, which were cracked significantly less than any other condition. After exhausting the Weir-algorithm guessing space in both experiments, basic16 remains significantly hardest to crack. Next best at resisting cracking were comprehensive8 and blacklistHard, performing significantly better than any other condition. Condition comprehensive8 was significantly better than blacklistHard in experiment P4 but not in experiment E. In experiment E, 14.6, 26.4, 31.0% of passwords were cracked in basic16, comprehensive8, and blacklistHard, respectively; in contrast, 63.0% of basic8 passwords were cracked.

Although guessing with the Weir algorithm proved more effective, we also compared the conditions using BFM. The findings (shown in Figure 3) are generally consistent with those discussed above: basic16 performs best.

In prior work examining memorability and usability for much of this dataset, we found that while in general less secure policies are more usable, basic16 is more usable than comprehensive8 by many measures [46]. This suggests basic16 is an overall better choice than comprehensive8.

It is important to note that 16-character-minimum policies are rare in practice. Hence, current guessing algorithms, including the Weir algorithm, are not built specifically with them in mind. Although we do not believe this affects our overall findings, it may merit further investigation.

B. *Effects of training-data selection*

Like most practical cracking algorithms, the ones we use rely on training data to determine guessing order. As a result, it is important to consider how the choice of training data affects the success of password guessing, and consequently the guess resistance of a set of passwords. To address this, we examine the effect of varying the amount and source of training data on both total cracking success and on cracking efficiency. Interestingly, we find that the choice of training data affects different password-policy conditions differently; abundant, closely matched training data is critical when cracking passwords from harder-to-guess conditions, but less so when cracking passwords from easier ones.

For purposes of examining the impact of training data, the password-policy conditions we consider divide fairly neatly into two groups. For the rest of this section, we will refer to the harder-to-guess conditions of comprehensive8, basic16, and blacklistHard as *group 1*, and the rest as *group 2*.

Training with general-purpose data. We first measure, via three experiments, the effect of increasing the amount and variety of training data. Experiment P3 was trained on public data including the MySpace and RockYou password lists as well as the inflection list and simple dictionary, and tested on 1000 passwords from each of our eight conditions. Experiment P4, as detailed in Section V-A, was trained on everything from P3 plus the paid Openwall list. Experiment E, also described in V-A, used everything from P4 plus 500 passwords from each of our conditions, using two-fold cross-validation. Figure 4 shows how these three training sets affect four example conditions, two from each group.

[3] All comparisons in Sections V-A, V-B, and V-C tested using PHFET, significance level $\alpha = 0.05$.

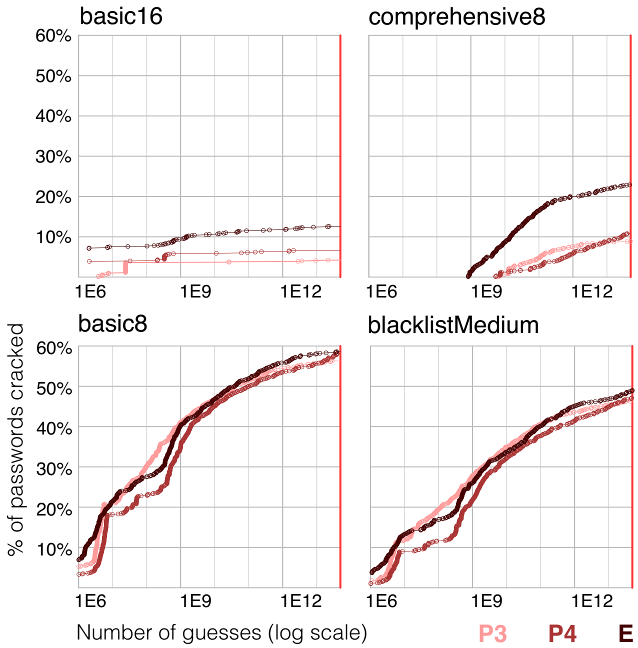

Figure 4. Showing how increasing training data by adding the Openwall list (P4) and then our collected passwords (E) affects cracking, for four example conditions. Adding training data proves more helpful for the group 1 conditions (top) than for the others (bottom).

As expected, cracking success increases as training data is added. For group 1, adding Openwall increases total cracking by 45% on average, while adding both Openwall and our data provides an average 96% improvement; these increases are significant for both experiments in all three conditions. In group 2, by contrast, the increases are smaller and only occasionally significant.

At one trillion and one billion guesses, the results are less straightforward, but increasing training data remains generally more helpful for cracking group 1 than group 2. Adding Openwall alone is not particularly helpful for group 1 conditions, with few significant improvements at either guessing point, but it actually decreases cracking at one billion guesses significantly for several group 2 conditions. (We hypothesize this decrease occurs because Openwall is a dictionary and not a password set, so it adds knowledge of structures and strings at the cost of accurately assessing their probabilities.) At these guessing points, adding our data is considerably more effective for group 1 than adding Openwall alone, increasing cracking for each of the three conditions by at least 50% (all significant). By contrast, adding our data provides little to no improvement against group 2 conditions at either guessing point.

Taken together, these results demonstrate that increasing the amount and variety of information in the training data provides significant improvement in cracking harder-to-guess conditions, while providing little benefit and sometimes decreasing efficiency for easier-to-guess conditions.

Training with specialized data. Having determined that training with specalized data is extremely valuable for cracking group 1 passwords, we wanted to examine what quantity of closely related training data is needed to effectively crack these "hard" conditions. For these tests, we focus on comprehensive8 as an example harder-to-guess condition, using the easier-to-guess basic8 condition as a control; we collected 3000 passwords each for these conditions.

In five Weir-algorithm experiments, C8a through C8e, we trained on all the public data from P4, as well as between 500 and 2500 comprehensive8 passwords, in 500-password increments. For each experiment, we tested on the remaining comprehensive8 passwords. We conducted a similar set of five experiments, B8a through B8e, in which we trained and tested with basic8 rather than comprehensive8 passwords.

Our results, illustrated in Figure 5, show that incrementally adding more of our collected data to the training set improves total cracking slightly for comprehensive8 passwords, but not for basic8. On average, for each 500 comprehensive8 passwords added to the training set, 2% fewer passwords remain uncracked. This effect is not linear, however; the benefit of additional training data levels off sharply between 2000 and 2500 training passwords. The differences between experiments begin to show significance around one trillion guesses, and increase as we approach the total number cracked.

For basic8, by contrast, adding more collected passwords to the training set has no significant effect on total cracking, with between 61 and 62% of passwords cracked in each experiment. No significant effect is observed at one million, one billion, or one trillion guesses, either.

One way to interpret this result is to consider the diversity of structures found in our basic8 and comprehensive8 password sets. The comprehensive8 passwords are considerably more diverse, with 1598 structures among 3000 passwords, as compared to only 733 structures for basic8. For comprehensive8, the single most common structure maps to 67 passwords, the most common 180 structures account for half of all passwords, and 1337 passwords have structures that are unique within the password set. By contrast, the most common structure in basic8 maps to 293 passwords, the top 13 structures account for half the passwords, and only 565 passwords have unique structures. As a result, small amounts of training data go considerably farther in cracking basic8 passwords than comprehensive8.

Weighting training data. The publicly available word lists we used for training are all considerably larger than the number of passwords we collected. As a result, we needed to weight our data (i.e., include multiple copies in the training set) if we wanted it to meaningfully affect the probabilities used by our guess-number calculators. Different weightings do not change the number of passwords cracked, as the same guesses will eventually be made; however, they can affect the order and, therefore, the efficiency of guessing.

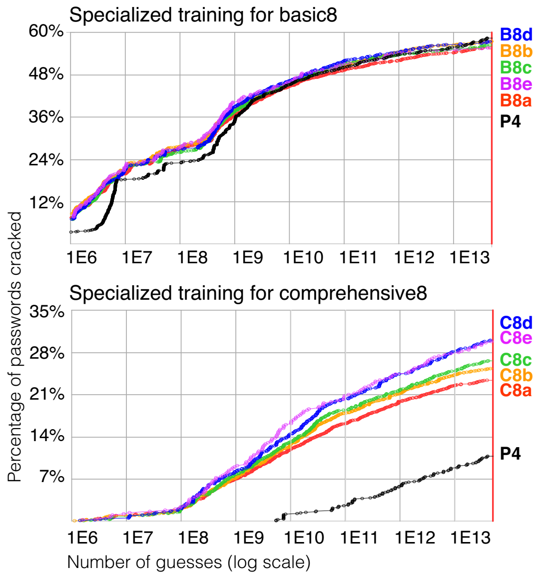

Figure 5. Top: Incremental increases in specialized training data have limited effect on the basic8 condition (B8a-B8e). Bottom: Incremental increases in specialized training data have a small but significant effect on the comprehensive8 condition (C8a-C8e). Results from P4 (the same public training data, no specialized training data) are included for comparison.

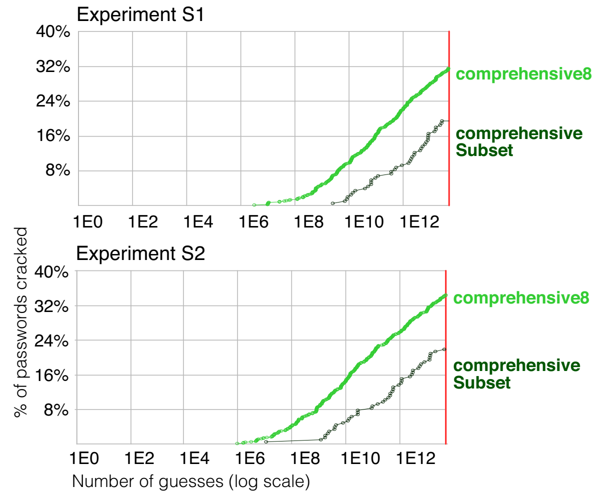

Figure 6. Passwords generated under the comprehensive8 condition proved significantly easier to guess than passwords that conform to the comprehensive8 requirements but are generated under other composition policies. In experiment S1 (top), the Weir calculator was trained with only public data; in experiment S2 (bottom), the Weir calculator was trained on a combination of our data and public data.

We tested three weightings, using 500 passwords from each condition weighted to one-tenth, equal, and ten times the cumulative size of the public lists. We tested each weighting on 500 other passwords from each condition.

Overall, we found that weighting had only a minor effect. There were few significant differences at one million, one billion, or one trillion guesses, with equal weighting occasionally outperforming the other two in some conditions. From these results, we concluded that the choice of weighting was not particularly important, but we used an equal weighting in all other experiments that train with passwords from our dataset because it provides an occasional benefit.

BFM training. We also investigated the effect of training data on BFM calculator performance, using four training sets: one with public data only, one that combined public data with collected passwords across our conditions, and one each specialized for basic8 and comprehensive8. Because the BFM algorithm eventually guesses every password, we were concerned only with efficiency, not total cracking. Adding our cross-condition data had essentially no effect at either smaller or larger numbers of guesses. Specialized training for basic8 was similarly unhelpful. Specialized training for comprehensive8 did increase efficiency somewhat, reaching 50% cracked with about 30% fewer guesses.

C. Effects of test-data selection

Researchers typically don't have access to passwords created under the password-composition policy they want to study. To compensate, they start with a larger set of passwords (e.g., the RockYou set), and pare it down by discarding passwords that don't meet the desired composition policy (e.g., [1], [13]). A critical question, then, is whether subsets like these are representative of passwords actually created under a specific policy. We find that such subsets are not representative, and may in fact contain passwords that are more resistant to guessing than passwords created under the policy in question.

In our experiments, we compared the guessability of 1000 comprehensive8 passwords to the guessability of the 206 passwords that meet the comprehensive8 requirements but were collected across our other seven conditions (the *comprehensiveSubset* set). We performed this comparison with two different training sets: public data, with an emphasis on RockYou passwords that meet comprehensive8 requirements (experiment S1); and the same data enhanced with our other 2000 collected comprehensive8 passwords (experiment S2).

Both experiments show significant differences between the guessability of comprehensive8 and comprehensiveSubset test sets, as shown in Figure 6. In the two experiments, 40.9% of comprehensive8 passwords were cracked on average, compared to only 25.8% comprehensiveSubset passwords. The two test sets diverge as early as one billion guesses (6.8% to 0.5%).

Ignoring comprehensiveSubset passwords that were created under basic16 leaves 171 passwords, all created under less strict conditions than comprehensive8. Only 25.2% of these are cracked on average, suggesting that subsets drawn exclusively from less strict conditions are more difficult to guess than passwords created under stricter requirements.

To understand this result more deeply, we examined the

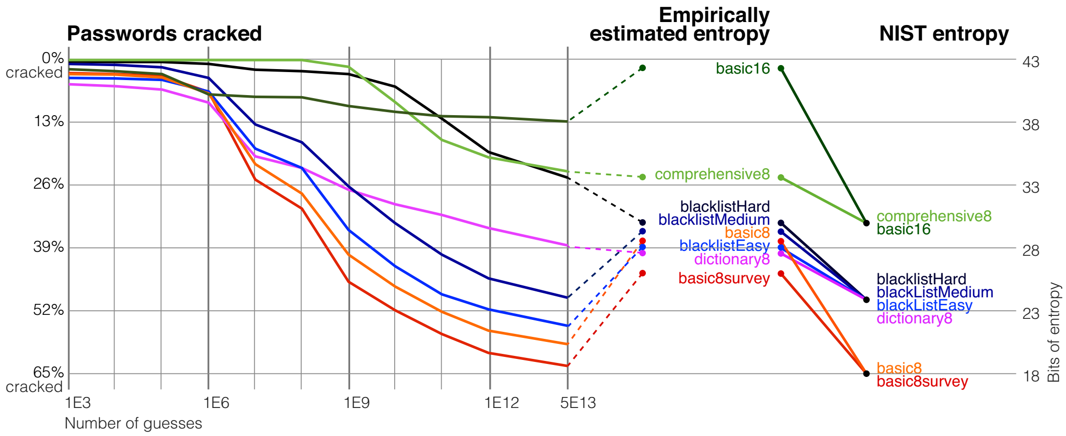

Figure 7. Relationship among the resistance of our collected password sets to heuristic cracking (experiment E); empirical entropy estimates we calculate from those sets; and NIST entropy estimates for our password conditions.

distribution of structures in the two test sets. There are 618 structures in the 1000-password comprehensive8 set, compared to 913 for comprehensiveSubset (normalized), indicating greater diversity in comprehensiveSubset passwords. This distribution of structures explains why comprehensive8 is significantly easier to guess.

We suspect this difference may be related to comprehensiveSubset isolating those users who make the most complex passwords. Regardless of the reason for this difference, however, researchers seeking to compare password policies should be aware that such subsets may not be representative.

D. Guessability and entropy

Historically, Shannon entropy (computed or estimated by various methods) has provided a convenient single statistic to summarize password strength. It remains unclear, however, how well entropy reflects the guess resistance of a password set. While information entropy does provide a theoretical lower bound on the guessability of a set of passwords [41], in practice a system administrator may be more concerned about how many passwords can be cracked in a given number of guesses than about the average guessability across the population. Although there is no mathematical relationship between entropy and this definition of guess resistance, we examine whether the two are correlated in practice. To do this, we consider two independent measures of entropy, as defined in Section IV-B: an empirically calculated estimate and a NIST estimate. For both measures, we find that entropy estimates roughly indicate which composition policies provide more guess resistance than others, but provide no useful information about the magnitude of these differences.

Empirically estimated entropy. We ranked our password conditions based on the proportion of passwords cracked in our most complete experiment (E) at one trillion guesses, and compared this to the rank of conditions based on empirically estimated entropy. We found these rankings, shown in Figure 7, to be significantly correlated (Kendall's $\tau = 0.71$, Holm-corrected $p = 0.042$). However, at one million or one billion guesses, the correlation in rankings is no longer significant (Holm-corrected $p = 0.275, 0.062$). We found the same pattern, correlation at one trillion guesses but not one billion or one million, in our largest public-data experiment (P4). These results indicate entropy might be useful when considering an adversary who can make a large number of guesses, but not when considering a smaller number of guesses.

Further, empirically estimated entropy did not predict the ranking of dictionary8, even when considering a large number of guesses. This condition displayed greater resistance to guessing than basic8, yet its empirically estimated entropy was lower. This might indicate a flaw in entropy estimation, a flaw in the guessing algorithm, or an innate shortcoming of the use of entropy to predict guessability. Since entropy can only lower-bound the guessability of passwords, it is possible for the frequency distribution of dictionary8 to have low entropy but high guess resistance. If this is the case, Verheul theorized that such a distribution would be optimal for password policy [51].

NIST entropy. Computing the NIST entropy of our password conditions produces three equivalence classes, as shown in Figure 7, because the heuristics are too coarse to capture all differences between our conditions. First, NIST entropy does not take into account the size of a dictionary or details of its implementation, such as case-insensitivity or removal of non-alphabetical characters before the check. All five of our dictionary and blacklist conditions meet the NIST requirement of a dictionary with at least 50,000 words [11]. Our results show that these variations lead to password policies with very different levels of password strength, which should be considered in a future heuristic.

Second, the NIST entropy scores for basic16 and comprehensive8 are the same, even though basic16 appears to be much more resistant to powerful guessing attacks. This may suggest that future heuristics should assign greater value to length than does the NIST heuristic.

Perhaps surprisingly, the equivalence classes given by NIST entropy are ordered correctly based on our results for guessability after 50 trillion guesses. Though it fails to capture fine-grained differences between similar password conditions, NIST entropy seems to succeed at its stated purpose of providing a "rough rule of thumb" [11].

We stress that although both measures of entropy provide a rough ordering among policies, they do not always correctly classify guessability (see for example dictionary8), and they do not effectively measure how much additional guess resistance one policy provides as compared to another. These results suggest that a "rough rule of thumb" may be the limit of entropy's usefulness as a metric.

VI. Discussion

We next discuss issues regarding ethics, ecological validity, and the limitations of our methodology.

Ethical considerations. Most of our results rely on passwords collected via a user study (approved by our institution's IRB). However, we also use the RockYou and MySpace password lists. Although these have collectively been used by a number of scientific works that study passwords (e.g., [2], [13], [25], [30]), this nevertheless creates an ethical conundrum: Should our research use passwords acquired illicitly? Since this data has already been made public and is easily available, using it in our research does not increase the harm to the victims. We use these passwords only to train and test guessing algorithms, and not in relationship with any usernames or other login information. Furthermore, as attackers are likely to use these password sets as training sets or cracking dictionaries, our use of them to evaluate password strength implies our results are more likely to be of practical relevance to security administrators.

Ecological validity. As with any user study, our results must be understood in context. As we describe in Section I, our participants are somewhat younger and more educated than the general population, but more diverse than typical small-sample password studies. The passwords we collected did not protect high-value accounts, reflecting a long-standing limitation of password research.

To further understand this context, we tested two password-creation scenarios (Section III-C): a survey scenario directly observing user behavior with a short-term, low-value account, and an email scenario simulating a longer-term, higher-value account. In both cases, users knew they might be asked to return and recall the password. Our users provided stronger passwords (measured by guessability and entropy) in the email scenario, a result consistent with users picking better passwords to protect a (hypothetical) high-value e-mail account than a low-value survey account.

To get real-world measures of password-related behavior, we surveyed users of Carnegie Mellon University's email system, which uses the comprehensive8 policy [9]. Comparing these survey results to the reports of our MTurk study participants, we find that on several measures of behavior and sentiment, the university responses (n = 280) are closer to those of our comprehensive8 participants than those of any other condition. For example, we asked MTurk participants who returned for the second half of the study whether they stored the password they had created (reassuring them they would get paid either way); we similarly asked university participants whether they store their login passwords. 59% of the university respondents report writing down their password, compared with 52% of comprehensive8 participants and a maximum of 37% for other MTurk conditions. These results show that study participants make different decisions based on password-composition requirements, and that in one condition their behavior is similar to people using that policy in practice.

We designed our study to minimize the impact of sampling and account-value limitations. All our findings result from comparisons *between* conditions. Behavior differences *caused by* the ways in which conditions differ (e.g., using a different technique to choose longer passwords than shorter ones) would be correctly captured and appropriately reflected in the results. Thus, we believe it likely that our findings hold in general, for at least some classes of passwords and users.

Other limitations. We tested all password sets with a number of password-guessing tools; the one we focus on (the Weir algorithm) always performed best. There may exist algorithms or training sets that would be more effective at guessing passwords than anything we tested. While this might affect some of our conclusions, we believe that most of them are robust, partly because many of our results are supported by multiple experiments and metrics.

In this work, we focused on automated offline password-guessing attacks. There are many other real-life threats to password security, such as phishing and shoulder surfing. Our analyses do not account for these. The password-composition policies we tested can induce different behaviors, e.g., writing down or forgetting passwords or using password managers, that affect password security. We report on some of these behaviors in prior work [46], but space constraints dictate that a comprehensive investigation is beyond the scope of this paper.

VII. Conclusion

Although the number and complexity of password-composition requirements imposed by systems administrators have been steadily increasing, the actual value added by these requirements is poorly understood. This work takes a substantial step forward in understanding not only these requirements, but also the process of evaluating them.

We introduced a new, efficient technique for evaluating password strength, which can be implemented for a variety of password-guessing algorithms and tuned using a variety of training sets to gain insight into the comparative guess resistance of different sets of passwords. Using this technique, we performed a more comprehensive password analysis than had previously been possible.

We found several notable results about the comparative strength of different composition policies. Although NIST considers basic16 and comprehensive8 equivalent, we found that basic16 is superior against large numbers of guesses. Combined with a prior result that basic16 is also easier for users [46], this suggests basic16 is the better policy choice. We also found that the effectiveness of a dictionary check depends heavily on the choice of dictionary; in particular, a large blacklist created using state-of-the-art password-guessing techniques is much more effective than a standard dictionary at preventing users from choosing easily guessed passwords.

Our results also reveal important information about conducting guess-resistance analysis. Effective attacks on passwords created under complex or rare-in-practice composition policies require access to abundant, closely matched training data. In addition, this type of password set cannot be characterized correctly simply by selecting a subset of conforming passwords from a larger corpus; such a subset is unlikely to be representative of passwords created under the policy in question. Finally, we report that Shannon entropy, though a convenient single-statistic metric of password strength, provides only a rough correlation with guess resistance and is unable to correctly predict quantitative differences in guessability among password sets.

VIII. Acknowledgments

We thank Mitch Franzos of the Carnegie Mellon Parallel Data Lab. We thank Stuart Schechter and Joe Bonneau for their feedback. This research was supported by NSF grants DGE-0903659, CNS-111676, and CCF-0424422; by CyLab at Carnegie Mellon under grant W911NF-09-1-0273 from the Army Research Office; and by Microsoft Research.

References

[1] D. Florêncio and C. Herley, "A large-scale study of web password habits," in *Proc. WWW'07*, 2007.

[2] M. Dell'Amico, P. Michiardi, and Y. Roudier, "Password strength: An empirical analysis," in *Proc. INFOCOM 2010*, 2010.

[3] M. Bishop and D. V. Klein, "Improving system security via proactive password checking," *Computers & Security*, vol. 14, no. 3, pp. 233–249, 1995.

[4] L. Constantin, "Sony stresses that PSN passwords were hashed," http://news.softpedia.com/news/Sony-Stresses-PSN-Passwords-Were-Hashed-198218.shtml, May 2011.

[5] P. Bright, "Anonymous speaks: The inside story of the HBGary hack," http://arst.ch/q6g, February 2011.

[6] J. Bonneau, "The Gawker hack: how a million passwords were lost," Dec. 2010, http://www.lightbluetouchpaper.org/2010/12/15/the-gawker-hack-how-a-million-passwords- were-lost/.

[7] P. Bright, "'Military meltdown Monday': 90k military usernames, hashes released," http://arstechnica.com/tech-policy/news/2011/07/military-meltdown-monday-90k-military-usernames-hashes-released.ars, 2011.

[8] S. Gaw and E. W. Felten, "Password management strategies for online accounts," in *Proc. SOUPS*, 2006.

[9] R. Shay, S. Komanduri, P. Kelley, P. Leon, M. Mazurek, L. Bauer, N. Christin, and L. Cranor, "Encountering stronger password requirements: user attitudes and behaviors," in *Proc. SOUPS '10*, 2010.

[10] L. St. Clair, L. Johansen, W. Enck, M. Pirretti, P. Traynor, P. McDaniel, and T. Jaeger, "Password exhaustion: Predicting the end of password usefulness," in *Proc. ICISS*, 2006.

[11] W. E. Burr, D. F. Dodson, and W. T. Polk, "Electronic authentication guideline," NIST, Tech. Rep., 2006.

[12] R. W. Proctor, M.-C. Lien, K.-P. L. Vu, E. E. Schultz, and G. Salvendy, "Improving computer security for authentication of users: Influence of proactive password restrictions," *Behavior Res. Methods, Instruments, & Computers*, vol. 34, no. 2, pp. 163–169, 2002.

[13] M. Weir, S. Aggarwal, M. Collins, and H. Stern, "Testing metrics for password creation policies by attacking large sets of revealed passwords," in *Proc. CCS*, 2010.

[14] J. Ross, L. Irani, M. S. Silberman, A. Zaldivar, and B. Tomlinson, "Who are the crowdworkers? Shifting demographics in Mechanical Turk," in *Proc. CHI EA*, 2010.

[15] P. G. Ipeirotis, "Demographics of Mechanical Turk," New York University, Tech. Rep. CeDER-10-01, 2010.

[16] M. Buhrmester, T. Kwang, and S. D. Gosling, "Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data?" *Perspectives on Psychological Science*, vol. 6, no. 1, pp. 3–5, 2011.

[17] J. S. Downs, M. B. Holbrook, S. Sheng, and L. F. Cranor, "Are your participants gaming the system? Screening Mechanical Turk workers," in *Proc. CHI*, 2010.

[18] A. Kittur, E. H. Chi, and B. Suh, "Crowdsourcing user studies with Mechanical Turk," in *Proc. CHI*, 2008.

[19] M. Toomim, T. Kriplean, C. Pörtner, and J. Landay, "Utility of human-computer interactions: toward a science of preference measurement," in *Proc. CHI*, 2011.

[20] S. Chiasson, A. Forget, E. Stobert, P. C. van Oorschot, and R. Biddle, "Multiple password interference in text passwords and click-based graphical passwords," in *Proc. CCS*, 2009.

[21] C. Kuo, S. Romanosky, and L. F. Cranor, "Human selection of mnemonic phrase-based passwords," in *Proc. SOUPS*, 2006.

[22] H. Wimberly and L. M. Liebrock, “Using fingerprint authentication to reduce system security: An empirical study,” in *Proc. IEEE Symposium on Security and Privacy*, 2011.
[23] D. Davis, F. Monrose, and M. K. Reiter, “On user choice in graphical password schemes,” in *Proc. USENIX Security Symposium*, 2004.
[24] A. Vance, “If your password is 123456, just make it hackme,” New York Times, http://nyti.ms/w8NNwD, January 2010.
[25] M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek, “Password cracking using probabilistic context-free grammars,” in *Proc. IEEE Symposium on Security and Privacy*, 2009.
[26] D. Hart, “Attitudes and practices of students towards password security,” *Journal of Computing Sciences in Colleges*, vol. 23, no. 5, pp. 169–174, 2008.
[27] A. Narayanan and V. Shmatikov, “Fast dictionary attacks on passwords using time-space tradeoff,” in *Proc. CCS*, 2005.
[28] M. Zviran and W. J. Haga, “Password security: an empirical study,” *J. Mgt. Info. Sys.*, vol. 15, no. 4, 1999.
[29] S. Komanduri and D. R. Hutchings, “Order and entropy in picture passwords,” in *Graphics Interface*, 2008.
[30] J. Bonneau, M. Just, and G. Matthews, “What’s in a name? Evaluating statistical attacks on personal knowledge questions,” in *Proc. Financial Crypto. 2010*, 2010.
[31] Y. Zhang, F. Monrose, and M. K. Reiter, “The security of modern password expiration: an algorithmic framework and empirical analysis,” in *Proc. CCS*, 2010.
[32] K.-P. L. Vu, R. W. Proctor, A. Bhargav-Spantzel, B.-L. B. Tai, and J. Cook, “Improving password security and memorability to protect personal and organizational information,” *Int. J. of Human-Comp. Studies*, vol. 65, no. 8, pp. 744–757, 2007.
[33] A. Adams, M. A. Sasse, and P. Lunt, “Making passwords secure and usable,” in *HCI 97*, 1997.
[34] P. Inglesant and M. A. Sasse, “The true cost of unusable password policies: password use in the wild,” in *Proc. ACM CHI’10*, 2010, pp. 383–392.
[35] R. Shay, A. Bhargav-Spantzel, and E. Bertino, “Password policy simulation and analysis,” in *ACM workshop on Digital identity management*, 2007, pp. 1–10.
[36] R. Shay and E. Bertino, “A comprehensive simulation tool for the analysis of password policies,” *Int. J. Info. Sec.*, vol. 8, no. 4, pp. 275–289, 2009.
[37] J. M. Stanton, K. R. Stam, P. Mastrangelo, and J. Jolton, “Analysis of end user security behaviors,” *Comp. & Security*, vol. 24, no. 2, pp. 124 – 133, 2005.
[38] D. Florêncio and C. Herley, “Where do security policies come from?” in *Proc. SOUPS ’10*, 2010.
[39] S. Schechter, C. Herley, and M. Mitzenmacher, “Popularity is everything: A new approach to protecting passwords from statistical-guessing attacks,” in *Proc. HotSec’10*, 2010.
[40] C. E. Shannon, “A mathematical theory of communication,” *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 1949.
[41] J. L. Massey, “Guessing and entropy,” in *Proc. IEEE Int. Symp. Info. Theory*, 1994, p. 204.
[42] C. Castelluccia, M. Dürmuth, and D. Perito, “Adaptive Password-Strength meters from markov models,” in *Proc. NDSS 2012*, 2012.
[43] S. Marechal, “Advances in password cracking,” *Journal in Computer Virology*, vol. 4, no. 1, pp. 73–81, 2008.
[44] C. M. Weir, “Using probabilistic techniques to aid in password cracking attacks,” Ph.D. dissertation, 2010.
[45] S. Designer, “John the Ripper,” http://www.openwall.com/john/, 1996-2010.
[46] S. Komanduri, R. Shay, P. G. Kelley, M. L. Mazurek, L. Bauer, N. Christin, L. F. Cranor, and S. Egelman, “Of passwords and people: Measuring the effect of password-composition policies,” in *Proc. CHI*, 2011.
[47] B. Schneier, “Myspace passwords aren’t so dumb,” http://www.wired.com/politics/security/commentary/securitymatters/2006/12/72300, December 2006.
[48] T. White, *Hadoop: The Definitive Guide*, 2nd ed. O’Reilly, September 2010.
[49] J. Dean and S. Ghemawat, “MapReduce: Simplified data processing on large clusters,” in *Proc. OSDI*, 2004.
[50] C. E. Shannon, “Prediction and entropy of printed english,” *Bell Systems Tech. J.*, vol. 30, 1951.
[51] E. Verheul, “Selecting secure passwords,” in *Topics in Cryptology–CT-RSA 2007*, 2006.

Appendix

Here we detail the complete training and test data used in each of our Weir-algorithm experiments. The first column gives the experiment number. The next three columns list the three types of training data used to create a Weir-calculator experiment. The *structures* column shows the wordlists used to generate the set of character-type structures that define the Weir algorithm’s search space. The wordlists in the *digits and symbols* column determine the probabilities for filling combinations of digits and symbols into these structures. The wordlists in the *strings* column determine the probabilities for filling alphabetic strings into structures. In most cases, we train strings on as much data as possible, while restricting structure and digit/symbol training to wordlists that contain a quality sample of multi-character-class passwords. The final column describes the password set(s) we attempted to guess.

We also list the complete training and test data used in each of our BFM experiments. The experiment number and test set columns are the same as in the Weir subtable. Training for the BFM calculator, however, uses only one combined wordlist per experiment; these lists are detailed in the *training set* column.

Abbreviations for all the training and test sets we use are defined in the key below the tables.

Weir experiment descriptions

Name	Training sets: Structures	Digits and symbols	Strings	Testing Set
Trained from public password data				
P1	MS8	MS	MS	1000-All
P2	MS8	MS	MS, W2, I	1000-All
P3	MS8	MS, RY	MS, W2, I, RY	1000-All
P3-C8	MSC	MS, RY	MS, W2, I, RY	1000-C8
P3-B16	MS16	MS, RY	MS, W2, I, RY	1000-B16
P4	MS8, OW8	MS, RY, OW	MS, W2, I, RY, OW	1000-All
P4-B16	MS16, OW16	MS, RY, OW	MS, W2, I, RY, OW	1000-B16
Trained on half of our dataset, weighted to 1/10th, equal-size, or 10x the cumulative size of the public data				
X1/10	MS8, 500-All	MS, RY, 500-All	MS, W2, I, RY, 500-All	500-All
X1	MS8, 500-All	MS, RY, 500-All	MS, W2, I, RY, 500-All	500-All
X10	MS8, 500-All	MS, RY, 500-All	MS, W2, I, RY, 500-All	500-All
Everything				
E	MS8, OW8, 500-All	MS, RY, OW, 500-All	MS, W2, I, RY, OW, 500-All	500-All
Testing password subsets that meet comprehensive8 requirements				
S0a	MSC, OWC	MS, OW	MS, W2, I, OW	1000-C8, 206-C8S
S0b	MSC, OWC, 2000-C8	MS, OW, 2000-C8	MS, W2, I, OW, 2000-C8	1000-C8, 206-C8S
S1	MSC, OWC, RYCD	MS, OW, RY	MS, W2, I, OW, RY	1000-C8, 206-C8S
S2	MSC, OWC, 2000-C8, RYCD	MS, OW, 2000-C8, RY	MS, W2, I, OW, 2000C8, RY	1000-C8, 206-C8S
Split ratio testing on basic8				
B8a	MS8, OW8, 500-B8	MS, RY, OW, 500-B8	MS, W2, I, RY, OW, 500-B8	2500-B8
B8b	MS8, OW8, 1000-B8	MS, RY, OW, 1000-B8	MS, W2, I, RY, OW, 1000-B8	2000-B8
B8c	MS8, OW8, 1500-B8	MS, RY, OW, 1500-B8	MS, W2, I, RY, OW, 1500-B8	1500-B8
B8d	MS8, OW8, 2000-B8	MS, RY, OW, 2000-B8	MS, W2, I, RY, OW, 2000-B8	1000-B8
B8e	MS8, OW8, 2500-B8	MS, RY, OW, 2500-B8	MS, W2, I, RY, OW, 2500-B8	500-B8
Split ratio testing on comprehensive8				
C8test1/10	MSC, 500-C8	MS, RY, 500-C8	MS, W2, I, RY, 500-C8	2500-C8
C8test1	MSC, 500-C8	MS, RY, 500-C8	MS, W2, I, RY, 500-C8	2500-C8
C8a	MSC, OWC, 500-C8	MS, RY, OW, 500-C8	MS, W2, I, RY, OW, 500-C8	2500-C8
C8b	MSC, OWC, 1000-C8	MS, RY, OW, 1000-C8	MS, W2, I, RY, OW, 1000-C8	2000-C8
C8c	MSC, OWC, 1500-C8	MS, RY, OW, 1500-C8	MS, W2, I, RY, OW, 1500-C8	1500-C8
C8d	MSC, OWC, 2000-C8	MS, RY, OW, 2000-C8	MS, W2, I, RY, OW, 2000-C8	1000-C8
C8e	MSC, OWC, 2500-C8	MS, RY, OW, 2500-C8	MS, W2, I, RY, OW, 2500-C8	500-C8

BFM experiment descriptions

Name	Training set	Test set
B1	RY, MS, I	1000-All
B2	RY, MS, I, 500-All	500-All
B3	RY, MS, I, 2000-B8	1000-B8
B4	RY, MS, I, 2000-C8	1000-C8

Key to password sets

RY	RockYou list	**I**	inflection list
RYCD	RY, filtered w/ all reqs. of C8	**W2**	simple Unix dictionary
MS	MySpace list	**OW**	paid Openwall dictionary
MS8	MS, filtered w/ min length of 8	**OW8**	OW, filtered w/ min length of 8
MS16	MS, filtered w/ min length of 16	**OW16**	OW, filtered w/ min length of 16
MSC	MS, filtered w/ min length of 8 and character class reqs. of C8	**OWC**	OW, filtered w/ min length 8 and character class reqs. of C8
n-**All**	n passwords from each of our conditions	n-**B8**	n basic8 passwords
n-**B16**	n basic16 passwords	n-**C8**	n comprehensive8 passwords
n-**C8S**	n comprehensiveSubset passwords	n-**RYCD**	n RYCD passwords