DOI:10.1145/2934663

Enterprises that impose stringent password-composition policies appear to suffer the same fate as those that do not.

BY DINEI FLORÊNCIO, CORMAC HERLEY, AND PAUL C. VAN OORSCHOT

Pushing on String: The 'Don't Care' Region of Password Strength

WE EXAMINE THE efficacy of tactics for defending password-protected networks from guessing attacks, taking the viewpoint of an enterprise administrator whose objective is to protect a population of passwords. Simple analysis allows insights on the limits of common approaches and reveals that some approaches spend effort in "don't care" regions where added password strength makes no difference. This happens either when passwords do more than enough to resist online attacks while falling short of what is needed against offline attacks or when so many accounts have fallen that an attacker gains little from additional compromises.

Our review of tools available to improve attack-resistance finds that, for example, compelling returns are offered by password blacklists, throttling, and hash iteration, while current password-composition policies fail to provide demonstrable improvement in outcomes against offline guessing attacks.

Suppose a system administrator is tasked with defending a corporate, government, or university network or site. The user population's passwords are targeted by attackers seeking access to network resources. Passwords can be attacked by guessing attacks—online or offline—and by capture attacks, that is, by non-guessing attacks. How to choose among password-attack mitigations is a practical question faced by millions of administrators. Little actionable guidance exists on how to do so in a principled fashion. Sensible policies must consider how to protect the population of user accounts. What is the best measure of the strength of a population of passwords? Is a good proxy for the overall ability of the network to resist guessing attacks the average, median, strongest, or weakest password? Slogans (such as suggesting that all passwords be "as strong as possible") are too vague to guide action—and also suggest that infinite user effort is both available and achievable.

» key insights

- It has long been accepted that making users choose more complex passwords is the price they must pay to make their accounts safer; it turns out this line of thinking misunderstands how attacks actually work.
- Harder-to-guess passwords do not always reduce the likelihood of successful guessing attacks; in fact, in a large portion of the attack space, they make no difference at all.
- Enterprises should focus on users with the most easily guessed passwords; an attacker probably gets all the access needed just by compromising them, so improving other passwords denies the attacker very little.

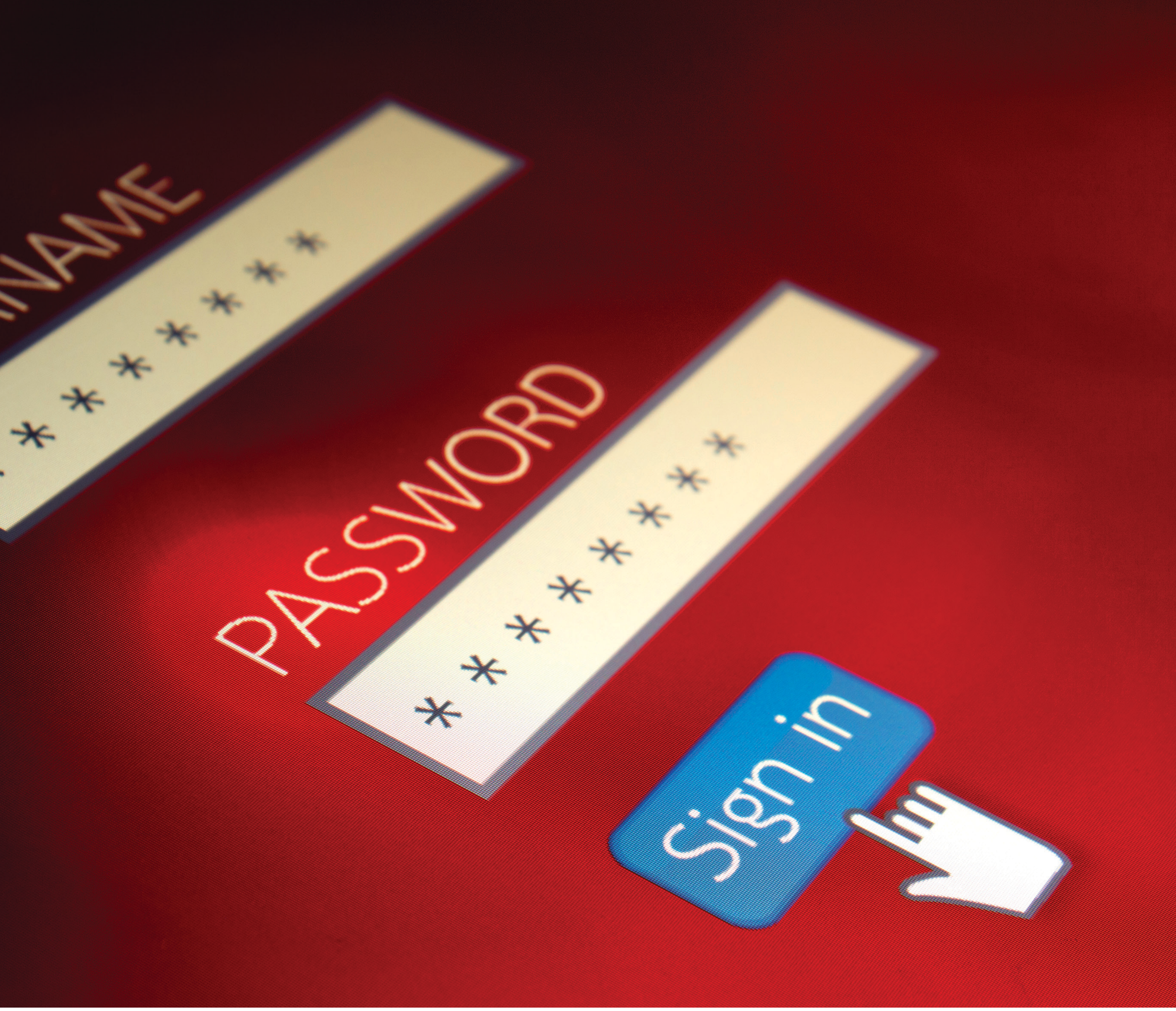


IMAGE BY RANGSAN PAIDAEN

The Compromise Saturation Point

To begin, consider an administrator who observes unusual behavior on his network and suspects that some accounts have been compromised. If he thinks it is just a few accounts, and can identify them, he might just block access to those. However, if he cannot be sure that an account is compromised until he sees suspicious activity, it is difficult to figure out the magnitude of the problem. Should he block access to all accounts and trigger a systemwide reset? If only a few accounts have been compromised, perhaps not; if half of them have been, he almost certainly should. What about a 1% compromise rate, or 5%, or 10%? At what point is global reset the right answer?

Suppose our hypothetical administrator resets all accounts. There still remains the question: How were the credentials obtained in the first place? If the door that led to the compromise remains open (such as undetected keyloggers on several machines) then nothing improves after a systemwide credential reset. On the other hand, if the credentials were compromised by guessing, then a reset (at least temporarily) helps, and a change in the policies that allowed vulnerable passwords might be in order. But even if he concludes that password guessing was the attack channel, was it online guessing? Or, somehow, did the attacker get hold of the password hash file and succeed with an offline guessing attack?

When attacks on passwords succeed, the specific attack channel is not necessarily clear; it is not obvious whether the compromised accounts had weak passwords, were spear-phished, or were drive-by download victims. Further, if it is not known which accounts have fallen, it may be best to reset all of them, even those not compromised. To facilitate more precise reasoning, let α be the fraction of credentials under attacker control—whether or not yet exploited. Thus, when $\alpha = 0.5$, half of the accounts (passwords) have already fallen to an attacker. At that point, would the administrator consider the network only 50% compromised, or fully overrun? The answer depends of course on the nature of the network in question. If a compromised account never has implications for any other account, then we might say the damage grows more-or-less linearly with α. However, for an enterprise network, a compromised account almost certainly has snowballing effects;[3] a single credential might give access to many network resources, so the damage to the network grows faster than α (one possible curve is shown in Figure 1). At $\alpha = 0.5$, a system is arguably completely overrun. For many enterprise environments in this scenario, there would be few if any resources the attacker cannot access; in many social networks, the network value would approach zero, as spam would probably render things unusable; access to (probably well under) 50% of email inboxes likely yields a view to almost all company email, as an attacker requires access to only one of the sender(s) or recipient(s) of each message.

All password-based systems must tolerate some compromise; passwords will be keylogged, cross-site scripted, and spear-phished, and a network unable to handle this reality will not be able to function in a modern threat environment. As an attacker gains more and more credentials in an enterprise network, she naturally reaches some saturation point after which the impact of additional fallen credentials is negligible, having relatively little effect on the attacker's ability to inflict harm. The first credential gives initial access to the network, and the second, third, and fourth solidify the beachhead, but the benefit brought by each additional credential decreases steadily. The gain is substantial when k is small, less when large; the second password guessed helps a lot more than, say, password 102.

Let α_{sat} be the threshold value at which the attacker effectively has control of the password-protected part of the system, in the sense that there is negligible marginal gain from compromising additional credentials. That is, if an attacker had control of a fraction α_{sat} of account credentials, there are very few resources that she could not access; so the difference between controlling α_{sat} and $(\alpha_{sat} + \varepsilon)$ is negligible. In what follows, our main focus is enterprise networks to consider possible values for α_{sat}.

There are a variety of tools attackers can use, once they have one set of credentials, to get others. Phishing email messages that originate from an internal account are far more likely to deceive co-workers. Depending on the attacker's objective, a toehold in the network may be all she requires. The 2011 attack on RSA[1] (which forced a recall of all SecurID tokens) began with phishing email messages to "two small groups of employees,"[13] none "particularly high-profile or high-value targets." Dunagan et al.[3] in examining a corporate network of more than 100,000 machines, found that 98.1% of the machines allowed an outward snowballing effect allowing compromise of 1,000 additional machines. Edward Snowden was able to compromise an enormous fraction of secrets on the NSA network starting from just one account.[14] Given that RSA and the NSA (organizations we might expect to have above-average security standards) experienced catastrophic failures caused by handfuls of credentials in attackers' hands, we suggest a reasonable upper bound on the saturation point for a corporate or government network is $\alpha_{sat} \approx 0.1$; saturation likely occurs at much lower values.

It seems likely that enterprises will have the lowest values of α_{sat}; at consumer Web services, compromise of one account has less potential to affect the whole network. As noted earlier, our focus here is on enterprise; nonetheless, we suggest that damage probably also grows faster than linearly at websites. For example, online accounts at a bank should have minimal crossover effects, but at 25% compromise all confidence in the legitimacy of transaction requests and the privacy of customer data is likely lost.

For guessing attacks, the most easily guessed passwords fall first. It is thus the weakest passwords that determine α_{sat}; the number of guesses it takes to gather a cumulative fraction α_{sat} of accounts is what it takes to reach the saturation point. Since the attacker's ability to harm saturates once she reaches α_{sat}, the excess strength of the remaining $(1\text{-}\alpha_{sat})$ of user passwords is wasted. For example, the strongest 50% of passwords might indeed be very strong, but from a systemwide viewpoint, that strength will come into play only when the other 50% of credentials has already been compromised—and

The argument—"stronger passwords are always better"—while deeply ingrained, thus appears untrue.

the attacker already has the run of everything that is password-protected in the enterprise network.

In summary, password strength, or guessing resistance, is not an abstract quantity to be pursued for its own sake. Rather, it is a tool we use to deny an attacker access to network resources. There is a saturation point, α_{sat}, where the network is so thoroughly penetrated that additional passwords gain the attacker very little; resistance to guessing beyond that point is wasted since it denies the attacker nothing. There is thus a "don't care" region after that saturation level of compromise, and for enterprise networks, it appears quite reasonable to assume that α_{sat} is no higher than 0.1. At that level, it is not simply the case that the weakest 10% of credentials is most important, but that the excess strength of the remaining 90% is largely irrelevant to the administrator's goal of systemwide defense. Of course there may be secondary benefits to individual users in having their password withstand guessing, even after α_{sat} has been exceeded. For example, if reused at another site, the user still has a significant stake in seeing the password withstand guessing. Since our focus is on the administrator's goal of protecting a single site, this is not part of our model.

The Online-Offline Chasm

We next consider the difference between online and offline guessing. In online attacks, an attacker checks guesses against the defender's server, that is, submitting guesses to the same server as legitimate users. In offline, she uses her own hardware resources, including networked or cloud resources and machines equipped with graphical processing units (GPUs) optimized for typical hash computations required to test candidate guesses. Offline attacks can thus test many-orders-of-magnitude more guesses than online attacks, whether or not online attacks are rate-limited by system defenses. We consider online and offline guessing attacks separately.

An online attack is always possible against a Web-facing service, assuming it is easy to obtain or generate valid account user IDs. An offline attack is possible only if the attacker gains access to the file of password hashes (in order to verify correctness of guesses offline), and, in this case, an offline attack is necessary only if that file has been properly salted and hashed; otherwise, simpler attacks are possible[6] (such as rainbow tables for unsalted hashed passwords).

There is an enormous difference between the strength required to resist online and offline guessing attacks. Naturally, the probability of falling either to an online or offline attack decreases gradually with the number of guesses a password will withstand. One hundred guesses per account might be easy for an online attacker, but 1,000 is somewhat more difficult, and so on; at some point, online guessing is no longer feasible. Similarly, at some point the risk from an offline attack begins to gradually decrease. Let T_0 be a threshold representing the maximum number of guesses expected from an online attack and T_1 correspondingly the minimum

Figure 1. The fraction of network resources under attacker control grows faster than the fraction of credentials compromised.

For example, given half of a system's credentials, an attacker likely effectively has access to all resources. For enterprise networks, we expect the saturation threshold, α_{sat}, where the network is completely penetrated, is likely under 0.1.

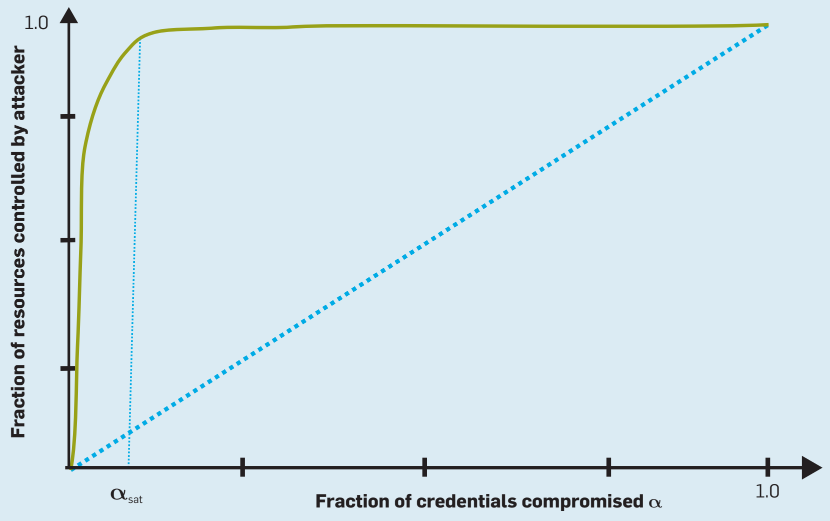
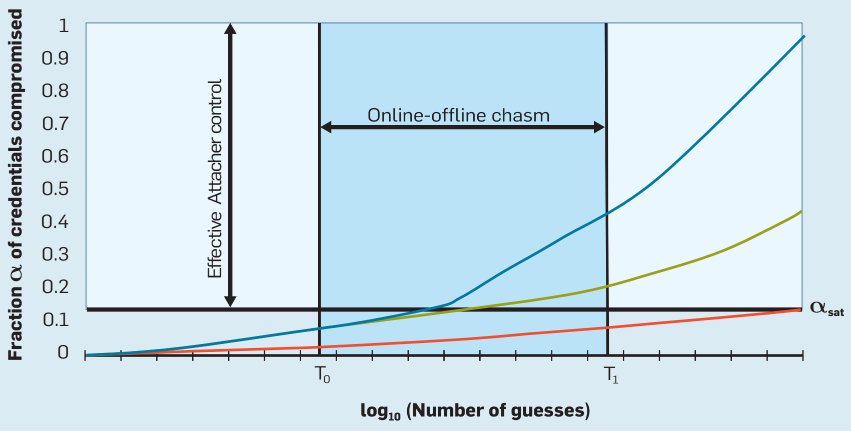


Figure 2. "Don't care" regions where there is no return for increasing effort.

T_0 is the threshold above which online attacks cease to be a threat.
T_1 is the threshold below which passwords almost surely will not survive credible offline attacks.
α_{sat} is the threshold fraction of compromised accounts at which an attacker effectively has control of system resources. Examples for these parameters might be $T_0 = 10^6$, $T_1 = 10^{14}$, and $\alpha_{sat} = 0.1$.



number of guesses expected from a credible offline attack. (The asymmetry in these definitions is intentional to provide a conservative estimate in reasoning about the size of the gap.) A password withstanding T_0 guesses is then safe from online guessing attacks, while one that does not withstand T_1 guesses will certainly not survive offline attacks. Our own previous work[6] suggests $T_0 \approx 10^6$ and $T_1 \approx 10^{14}$ are reasonable coarse estimates, giving a gap eight orders of magnitude wide (see Figure 2); we emphasize, however, that the arguments herein are generic, regardless of the exact values of T_0 and T_1. While estimates for T_1 in particular are inexact, depending as they do on assumptions about attacker hardware and strategy (a previous estimate[6] assumed four months of cracking against one million accounts using 1,000 GPUs, each doing one billion guesses per second), clearly T_1 vastly exceeds T_0.

A critical observation is that for a password *P* whose guessing-resistance falls between T_0 and T_1, incremental effort that fails to move it beyond T_1 is wasted in the sense that there is no guessing attack to which the original *P* falls, that the stronger password resists, since guessing attacks are either online or offline, with no continuum in between. Passwords in this online-offline gap are thus in a "don't care" region in which they do both too much and not enough—too much if the attack vector is online guessing and not enough if it is offline. Once a password is able to withstand T_0 guesses, to stop additional attacks, any added strength must be sufficient to move to the right of T_1. While both online and offline attacks are countered by guessing-resistance, the amount needed varies enormously. In practical terms, distinct defenses are required to stop offline and online attacks. This online-offline chasm then gives us a second "don't care" region, besides the one defined by α_{sat}.

The "Don't Care" Region

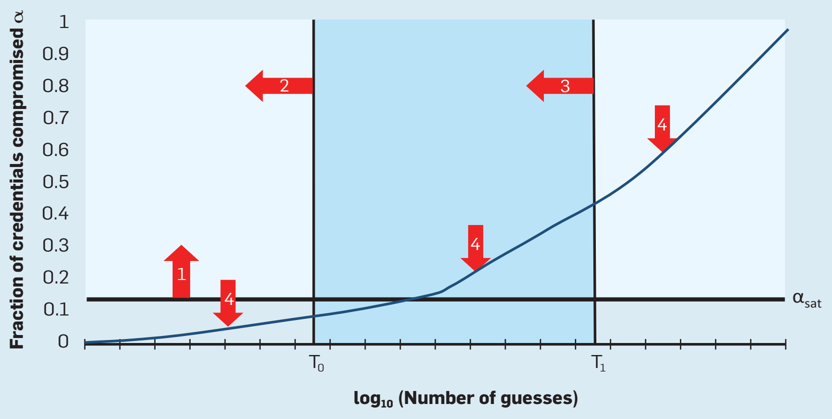
The compromise saturation point and the online-offline chasm each imply regions where there is no return on effort. The marginal return on effort is zero for improving any password that starts in the zone greater than T_0 yet remains less than T_1 and for passwords with guessing-resistance above the α_{sat} threshold. Figure 2 depicts this situation, with shaded areas denoting the "don't care" regions. A password withstanding $T_1 - \varepsilon$ guesses has the same survival properties as one surviving $T_0 + \varepsilon$; both survive online but fall to offline attack, that is, equivalent outcomes. From the administrator's point of view, the strongest password in the population, or having the greatest guessing-resistance (the password at $\alpha = 1.0$) is similar to one at $\alpha_{sat} + \varepsilon$, in that both fall after the attacker's ability to harm is already saturated.

In summary, shaded regions denote these areas where there is no return-on-effort for "extra strength": first, passwords whose guessing-resistance lies within the online-offline gap; and second, passwords beyond where an attacker gains little from additional credentials. The size of the "don't care" region naturally depends on the particular values of α_{sat}, T_0, and T_1, but in all cases the shape of the password distribution (as defined by the colored curves in Figure 2) matters only in the areas below α_{sat} AND (left of T_0 OR right of T_1). An important observation is that, under reasonable assumptions, the "don't care" regions cover a majority of the design space. The relatively small unshaded regions are shown in Figure 2; outside these regions, changes to the password distribution accomplish nothing, at least from the administrator's viewpoint. To anchor the discussion, based on what we know of enterprise networks and attacker abilities, we have offered estimates of $\alpha_{sat} = 0.1$, $T_0 = 10^6$, and $T_1 = 10^{14}$; but choosing different values does not alter the conclusion that in large areas of the guess-resistance vs. credentials-compromised space, changing the distribution of user-chosen passwords improves little of use for the enterprise defender; it causes no direct damage but, like pushing on string, is ineffective and wasteful of user energy.

To understand the consequences of the "don't-care" regions, Figure 2 also depicts guessing resistance of three hypothetical password distributions. The blue (top) and green (middle) curves diverge widely on the right side of the figure; for a fixed number of guesses, far fewer green-curve accounts will be compromised than accounts from the blue-curve distribution. The green curve might appear better since those passwords are much more guess-resistant than those from the blue curve. Nonetheless, they have identical attack survival outcomes since their divergence between T_0 and T_1 has minimal

Figure 3. Defensive elements aiming to improve guessing resistance.

(1) α_{sat} (the point at which attacker control saturates) can be raised by implementing basic security principles (such as least-privilege and compartmentalization).
(2) T_0 (the maximum number of online guesses) can be reduced by throttling mechanisms.
(3) T_1 (the minimum number to be expected from an offline attack) can be reduced by iterated hash functions.
(4) The cumulative fraction of accounts that have fallen at a given number of attacker guesses can be reduced (pushing the blue curve down) by improving the guessing-resistance of user-chosen passwords (such as to the left of T_0 by password blacklisting and by password composition policies generally).
Changing α_{sat}, T_0 and T_1 alter the size of white- and blue-shaded "don't care" regions.



effect on performance against online or offline attacks, and divergence above α_{sat} happens only after the attacker's capacity for harm has already plateaued. The red (lower) curve shows a distribution that might survive an offline attack, as the curve lies below α_{sat}, even at the number of guesses an offline attacker might deliver. The enormous difficulty of getting users to choose passwords that will withstand offline guessing, and the waste that results unless almost all of them do, has also been argued by Tippett.[11] We reemphasize that α_{sat}, T_0, and T_1 are site- and implementation-dependent variables. We next examine how an administrator can vary them to decrease the size of the "don't care" zone.

What Should an Administrator Optimize?

Ideally, a system's population of passwords would withstand both online and offline attacks. For this to be so, the fraction of accounts compromised must be lower than α_{sat} at T_1 guesses, and ideally much lower, since an offline attacker may be able to go well beyond the expected minimum T_1. Unfortunately, recent work shows that user-chosen passwords do not even approach this level, even when stringent composition policies are enforced. For example, Mazurek et al.[10] found that 48% of Carnegie Mellon University passwords (following a length-eight and four-out-of-four character sets policy) fell within 10^{14} guesses. On examining eight different password-creation policies, Kelley et al.[8] found that none kept the cumulative fraction of accounts compromised below 10% by 10^{11} guesses. If we believe an attacker's control saturates by the time a fraction $\alpha_{sat} = 0.1$ of accounts is compromised, and an offline attack can mount at least $T_1 = 10^{11}$ guesses per account, there is thus little hope of resisting offline attack. That is, with these assumed values of α_{sat} and T_1, an attacker who gains access to the hashed password file will have all the access she needs, no matter how far outside her reach the remaining fraction ($1-\alpha_{sat}$) of passwords lie.

What then should an organization seek to do? It is ideal if passwords robustly withstand both online and offline attacks, as is the case for the lower curve in Figure 2. However, striving for such resistance but falling short wastes user effort and accomplishes nothing. In Figure 2, all effort above and beyond that necessary to withstand online attack is completely wasted in the two upper curves. An organization that tries to withstand offline attacks but fails to have at least a fraction $1 - \alpha_{sat}$ of users survive T_1 guesses fares no better than one that never made the attempt and does worse if we assume user effort is a scarce resource.[5] The evidence of recent work on cracking on real distributions[8,10] suggests this is the fate of more or less all organizations that allow user-chosen passwords, unless we believe that $\alpha_{sat} \approx 0.1$ is unreasonably low or T1 $\approx 10^{14}$ is unreasonably high. The argument—"stronger passwords are always better"—while deeply ingrained, thus appears untrue. Stronger passwords lower the cumulative fraction of accounts compromised at a given number of guesses, that is, push the curves in Figure 2 lower. However, changes that occur within the shaded "don't care" regions happen when it no longer matters and do not improve outcomes.

Password strength, which actually means guessing resistance, is not a universal good to be pursued for its inherent benefits; it is useful only to the extent it denies things to adversaries.

It follows that a reasonable objective is to maximize, within reasonable costs, guessing-resistance across a given system's set of passwords at the expected online and offline number of guesses, subject to the constraint that the fraction of compromised accounts stays below α_{sat}. That is, so long as $\alpha < \alpha_{sat}$, the lower α is at T_0 (respectively T_1) the better the resistance to online (respectively offline) attacks. For $\alpha > \alpha_{sat}$, improvements that do not reduce α below α_{sat} are unrewarded. Focusing as our model does on a single site, we remind readers that the additional benefit of withstanding guessing to users who have reused their password at other sites is not captured in this model.

What Can an Administrator Control?

What tools does an administrator have to reach these goals? The outcome of any administrator actions will be influenced by the values α_{sat}, T_0, T_1, and the shape of the cumulative password distribution. We show these various forces in Figure 3.

The value of the compromise saturation point α_{sat} is largely determined by the network topology and might be relatively difficult to control or change in a given environment. Basic network hygiene and adherence to security principles (such as least privilege, isolation, and containment) can help minimize damage when intrusion occurs. These defenses are also effective against intrusions that do not involve password guessing. We assume these defenses will already be in force and in the rest of this section concentrate on measures that mostly affect password-guessing attacks.

Improving T_0. There are few, if any, good reasons for an authentication system to allow hundreds of thousands of distinct guesses on a single account. In cases of actual password forgetting, it is unlikely that the legitimate user types more than one dozen or so distinct guesses. Mechanisms that limit the number of online guesses (thus reducing T_0) include various throttling mechanisms (rate limiting) and IP address blacklisting. The possibility of denial-of-service attacks can usually be dealt with by IP address whitelisting. (We mean, not applying the throttling triggered by new IP addresses to known addresses from which a previous login succeeded; wrong guesses from that known address should still be subject to throttling.) A simple, easily implemented throttling mechanism may suffice for many sites. When denial-of-service attacks are a possibility, more complex mechanisms may be necessary, perhaps including IP address white- and blacklisting, and methods requiring greater effort from site administrators. Such defensive improvements should come without additional burdens of effort and inconvenience to users. Together with password blacklisting (discussed in the following section), throttling may almost completely shut down generic online guessing attacks.

Improving T_1. A password must withstand a relatively large number of guesses to have any hope of withstanding credible offline attacks. A lower bound T_1 on this number may vary depending on the defenses put in place and can be quite high. For example, if an attacker can make 10 billion guesses per second on each of 1,000 GPUs,[7] then in a four-month period, she can try approximately $T_1 = 10^{14}$ guesses against each of one million accounts.[6] Furthermore, technology advances typically aid attackers more than defenders; few administrators replace hardware every year, while attackers can be assumed to have access to the latest resources. This moves T_1 to the right—as computing speeds and technology advance and customized hardware gives attackers further advantages. Since the hardware is controlled by the attacker, little can be done to directly throttle offline guessing. However, an effective way to reduce the number of trials per second is to use a slow hash, that is, one that consumes more computation. The most obvious means is hash iteration;[12] recent research is also exploring the design of hash functions specifically designed to be GPU unfriendly. For example, ignoring the counteracting force of speed gains due to advancing technology, an iteration count of $n = 10^4$ reduces T_1 from 10^{14} to 10^{10}. Even with iteration it is difficult to move T_1 all the way left to T_0 to make the online-offline chasm disappear. Limits on further increasing n arise from the requirement that the time to verify legitimate users must be tolerable to both the users (wait time) and system hardware; for example, n might allow verifying 100 legitimate users/second (10ms per user). If 10ms is a tolerable delay, an attacker with access to 1,000 GPUs can compute a total of $1{,}000 \times 4 \times 30 \times 24 \times 60 \times 60/10^{-2} \approx 10^{12}$ guesses in four months. Directing this effort at 100 accounts would mean each would have to withstand a minimum of $T_1 = 10^{10}$ guesses. Since these are conservative assumptions, it appears challenging for a typical enterprise to decrease T_1 below this point.

Note that, as technology evolves, the number of hash iterations can easily be increased, invisibly to users and on the fly—by updating the stored password hashes in the systemside file to reflect new iteration counts.[12] Among the appealing aspects of iterated hashing, it is long known as an effective defensive tool, and costs are borne systemside rather than by user effort. However, hash iteration is not a miracle cure-all; for a password whose guess-resistance is 10^6, online throttling is still important. An online attacker who could test one password every 10ms (matching the system rate noted earlier) will succeed in 10^4 seconds = 2 hours, 47minutes.

If the hashed password file leaks, burdening users with complex composition policies does not alter the fact that a competent attacker likely gets access to all the accounts she could possibly need.

Eliminating offline attacks altogether. The emphasis by any parties who encourage users to choose stronger passwords can obscure the fact that offline attacks are only a risk when the password hash file "leaks," a euphemism for "is somehow stolen," or otherwise becomes available to an attacker. Any means or mechanisms that prevent the password hash file from leaking entirely remove the need for individual passwords to withstand an offline attack. Since we defined T_1 as the minimum number of guesses a password must withstand to resist an offline attack, any such mechanism effectively reduces T_1 to zero. We next discuss one particularly appealing such mechanism.

Hardware security modules (HSMs). A properly used HSM eliminates the risk of a hash file leaking[6] or, equivalently, eliminates the risk of a decryption key (or backup thereof) leaking in the case that the means used to protect the information stored systemside to verify passwords is reversible encryption. In such a proper HSM architecture, rather than a file of the one-way hashes of salted passwords, what is stored in each file entry is a message authentication code (MAC) computed over the corresponding password using a secret key. When a password candidate is presented for verification, the candidate plus the corresponding MAC from the system file are provided as HSM inputs. The HSM holds the system secret key used to compute the MAC; importantly, this secret key is by design never available outside the HSM. Upon receiving the (MAC, candidate password) input pair, the HSM independently computes a MAC over the input candidate, compares it to the input MAC, and answers yes (if they agree) or no if they do not. Stealing the password hash file—in this case a password MAC file—is now useless to the offline attacker, because the HSM is needed to verify guesses; that is, offline attacks are no longer possible.

Another interesting scheme to mitigate offline attacks, proposed by Crescenzo et al.,[2] bounds the number of guesses that can be made by restricting the bandwidth of the connection between the authentication server and a specially constructed hash server that requires a very large collection of random bits. This approach limits an attacker to online guessing, since, by design, the connection is too small to support the typical guessing rate an offline attacker needs, or to allow export of any file that would be useful to an offline attacker.

Improving the password distribution. Finally, we consider changes to the password distribution as a means of improving outcomes. Recall the curves in Figure 2 represent the cumulative fraction of accounts compromised as a function of the number of guesses per account. In general, it has been accepted without much second thought that the lower this cumulative fraction the better; a great deal of effort has gone into coercing users to choose supposedly "stronger" passwords, thus pushing the cumulative distribution curve downward in one or more of the three regions induced by T_0 and T_1. However, as explained earlier, lower is of tangible benefit only outside the "don't care" region; improvements to the curve inside the "don't care" region have negligible effects on outcomes in any attack scenario.

First, note that tools to influence the cumulative distribution are mostly indirect; users choose passwords, not administrators. For example, by some combination of education campaigns, password policies, and password meters, administrators may try to influence this curve toward "better" passwords. However, the cumulative distribution is ultimately determined by user password choice; if users ignore advice, do the minimum to comply with policies, or are not motivated by meters, then efforts to lower the curve may have little impact on user choices.

Second, note that many policy and education mechanisms are unfocused in the sense that they cannot be targeted at the specific part of the cumulative distribution where they make the most difference—and away from the "don't care" region where they make none. Even if they succeed, exhortations to "choose better passwords" are not concentrated at one part of the curve or another; if all users respond to such a request by improving their passwords marginally, the related effort of 90% of users is still wasted for an enterprise where $\alpha_{sat} = 0.1$. We now examine common approaches to influencing password choices in this light.

Password blacklisting. Attackers exploit the fact that certain passwords are common. Explicitly forbidding passwords known to be common can thus reduce risk. Blacklists concentrate at the head of the distribution, blocking the choice of most common passwords. For example, a user who attempts to choose "abcdefg" is informed this is not allowed and asked to choose another. Certain large services do this; for example, Twitter blacklisted 380 common passwords after an online guessing incident in 2009, and Microsoft applied a blacklist of several hundred to its online consumer properties in 2011. With improvements of password crackers and the recent wide availability of passwords lists, blacklists need to be longer.

A blacklist of, say, 10^6 common passwords may help bring guessing resistance to the 10^5 level. A natural concern with blacklists is that users may not understand why particular choices are forbidden. Kelley et al.[8] examined the guessing resistance of blacklists of various sizes, but the question of how long one can be before the decisions appear capricious is an open one. Komanduri et al.[9] pursue this question with a meter that displays, as a user types, the most likely password completion. A further unknown is the improvement achieved when users are told their password choice is forbidden. It appears statistically unlikely that all of the users who initially selected one of Twitter's 380 blacklisted passwords would collide again on so small a list when making their second choice, but we are not aware of any measurements of the dispersion achieved. A promising recent practical password strength estimator is `zxcvbn`,[15] a software tool that can also be of use for password blacklisting.

Observe that blacklists are both direct and focused: they explicitly prevent choices known to be bad rather than rely on indirect measures, and they target those users making bad choices, leaving the rest of the population unaffected. A blacklist appears to be one of the simplest measures to meaningfully improve the distribution in resisting online attacks.

Composition policies. Composition policies attempt to influence user-chosen passwords by mandating a minimum length and the inclusion of certain character types; a typical example is "length at least eight characters, and use three of the four character sets {lowercase, uppercase, digits, special characters}." Certain policies may help improve guess-resistance in the 10^5 to 10^8 range. However, for what we have suggested as the reasonable values $\alpha_{sat} = 0.1$ and $T_1 = 10^{14}$, the evidence strongly suggests that none of the password-composition policies in common use today or seriously proposed[8,10] can help; for these to become relevant, one must assume that an attacker's ability to harm saturates much higher than $\alpha_{sat} = 0.1$ or that the attacker can manage far fewer than $T_1 = 10^{14}$ offline guesses. Such policies thus fail to prevent total penetration of the network. Their ineffectiveness is perhaps the reason why a majority of large Web services avoid onerous policies.[4]

Note that composition policies are indirect: the constraints they impose are not themselves the true end objectives, but it is hoped they result in a more defensive password distribution. This problem is compounded by the fact that whether the desired improvement is indeed achieved is concealed from an administrator. The justifiably recommended practice of storing passwords as salted hashes means the password distribution is obscured, as are any improvements caused by policies. Composition policies are also unfocused in that they affect all users rather than being directed specifically where they may matter most. A policy may greatly affect user password choice and still have little effect on outcome (such as if all of the change in the cumulative distribution happens inside the "don't care" region).

Conclusion

Password strength, which actually means guessing resistance, is not a universal good to be pursued for its inherent benefits; it is useful only to the extent it denies things to adversaries. When we consider a population of accounts, there are large areas where increased guessing resistance accomplishes nothing—either because passwords fall between the online and offline thresholds or because so many accounts have already fallen that attacker control has already saturated. If increases in password guessing resistance were free this would not matter, but such increases are typically achieved at great cost in user effort; for example, there is a void of evidence that current approaches based on password composition policies significantly improve defensive outcomes and strong arguments that they waste much user effort. This situation thus creates risk of a false sense of security.

It is common to assume that users must choose passwords that will withstand credible offline attacks. However, if we assume an offline attacker can mount $T_1 = 10^{14}$ guesses per account and has all the access she needs by the time she compromises a fraction $\alpha_{sat} = 0.1$ of accounts, we must acknowledge that trying to stop offline attacks by aiming user effort toward choosing "better passwords" is unachievable in practice. The composition policies in current use seem so far from reaching this target that their use appears misguided. This is not to say that offline attacks are not a serious threat. However, it appears that enterprises that impose stringent password-composition policies on their users suffer the same fate as those that do not. If the hashed password file leaks, burdening users with complex composition policies does not alter the fact that a competent attacker likely gets access to all the accounts she could possibly need. Nudging users in the "don't care" region (where most passwords appear to lie) is simply a waste of user effort.

The best investments to defend against offline attacks appear to involve measures transparent to users. Iteration of password hashes lowers the T_1 boundary; however, even with very aggressive iteration, we expect that at least 10^{10} offline guesses remain quite feasible for attackers. Use of MACs, that is, keyed hash functions, instead of regular (unkeyed) password hashes, provides effective defense against offline attacks that exploit leaked hash files—provided the symmetric MAC key is not also leaked. Use of HSMs is one method of protecting MAC keys, as discussed; though more expensive than software-only defenses, HSMs can eliminate offline attacks entirely. Online guessing attacks, in contrast, cannot be entirely eliminated, but effective defenses include password blacklists and throttling. There appear few barriers to implementing these simple defenses. **C**

References

1. Bright, P. RSA finally comes clean: SecurID is compromised. *Ars Technica* (June 6, 2011); http://arstechnica.com/security/news/2011/06/rsa-finally-comes-clean-securid-is-compromised.ars/
2. Crescenzo, G.D., Lipton, R.J., and Walfish, S. Perfectly secure password protocols in the bounded retrieval model. In *Proceedings of the Theory of Cryptography Conference* (New York, Mar. 4–7). Springer-Verlag, 2006, 225–244.
3. Dunagan, J., Zheng, A.X., and Simon, D.R. Heat-Ray: Combating identity snowball attacks using machine learning, combinatorial optimization and attack graphs. In *Proceedings of the ACM Symposium on Operating Systems Principles* (Big Sky, MT, Oct. 11–14). ACM Press, New York, 305–320.
4. Florêncio, D. and Herley, C. Where do security policies come from? In *Proceedings of the SOUPS Symposium On Usable Privacy and Security* (Redmond, WA, July 14–16, 2010).
5. Florêncio, D., Herley, C., and van Oorschot, P. Password portfolios and the finite-effort user: Sustainably managing large numbers of accounts. In *Proceedings of the 23rd USENIX Security Symposium* (San Diego, CA, Aug. 20–22). USENIX Association, Berkeley, CA, 2014, 575–590.
6. Florêncio, D., Herley, C., and van Oorschot, P.C. An administrator's guide to Internet password research. In *Proceedings of the USENIX LISA Conference* (Seattle, WA, Nov. 9–14). USENIX Association, Berkeley, CA, 2014, 35–52.
7. Goodin, D. Why passwords have never been weaker and crackers have never been stronger. *Ars Technica* (Aug. 20, 2012); http://arstechnica.com/security/2012/08/passwords-under-assault/
8. Kelley, P.G., Komanduri, S., Mazurek, M.L., Shay, R., Vidas, T., Bauer, L., Christin, N., Cranor, L.F., and Lopez, J. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In *Proceedings of the IEEE Symposium on Security and Privacy* (San Francisco, May 20–23). IEEE Press, 2012, 523–537.
9. Komanduri, S., Shay, R., Cranor, L.F., Herley, C., and Schechter, S. Telepathwords: Preventing weak passwords by reading users' minds. In *Proceedings of the 23rd USENIX Security Symposium* (San Diego, CA, Aug. 20–22). USENIX Association, Berkeley, CA, 2014, 591–606.
10. Mazurek, M.L., Komanduri, S., Vidas, T., Bauer, L., Christin, N., Cranor, L.F., Kelley, P., Shay, R., and Ur, B. Measuring password guessability for an entire university. In *Proceedings of the 20th ACM Conference on Computer and Communications Security* (Berlin, Germany, Nov. 4–8). ACM Press, New York, 2013.
11. Tippett, P. Stronger passwords aren't. *Information Security Magazine* (June 2001), 42–43.
12. Provos, N. and Mazières, D. A future-adaptable password scheme. In *Proceedings of the 1999 USENIX Annual Technical Conference, FREENIX Track* (Monterey, CA, June 6–11). USENIX Association, Berkeley, CA, 1999, 81–91.
13. RSA FraudAction Research Labs. Anatomy of a hack. RSA, Bedford, MA, Apr. 1, 2011; https://blogs.rsa.com/anatomy-of-an-attack/
14. Toxen, B. The NSA and Snowden: Securing the all-seeing eye. *Commun. ACM 57*, 5 (May 2014), 44–51.
15. Wheeler, D. zxcvbn: Low-budget password strength estimation. In *Proceedings of the 25th USENIX Security Symposium* (Austin, TX, Aug. 10–12). USENIX Association, Berkeley, CA, 2016.

Dinei Florêncio (dinei@microsoft.com) is a senior researcher in the Multimedia and Interactive Experiences group of Microsoft Research, Redmond, WA.

Cormac Herley (cormac@microsoft.com) is a principal researcher at Microsoft Research, Redmond, WA.

Paul C. van Oorschot (paulv@scs.carleton.ca) is a professor of computer science and Canada Research Chair in Authentication and Computer Security at Carleton University, Ottawa, Canada.