Rethinking Password Policies

ABE SINGER AND WARREN ANDERSON

Abe Singer is the Chief Security Officer for the Laser Interferometer Gravitational Wave Observatory at the California Institute of Technology. He has been a programmer, system administrator, security geek, occasional consultant, and expert witness. His areas of interests are in security that actually works.
abe@ligo.caltech.edu

Warren Anderson is a Visiting Assistant Professor in the Department of Physics at the University of Wisconsin-Milwaukee and a member of the LIGO Scientific Collaboration. His publications are primarily on black holes and gravitational waves. This is his first foray into the fascinating world of computer security.
warren.anderson@ligo.org

We are all familiar with having "rules" for passwords: they must have characters from various character sets, have a minimum length, get changed regularly, not be written down, etc. These rules are supposed to make passwords "secure," but there's little to no research to support that argument. In fact, they can even weaken security. We argue that it's time for a radical change of password policy. In the blog post "Security Myths and Passwords," Gene Spafford also made the case for questioning the conventional wisdom on security:

> In the practice of security we have accumulated a number of "rules of thumb" that many people accept without careful consideration. Some of these get included in policies, and thus may get propagated to environments they were not meant to address. It is also the case that as technology changes, the underlying (and unstated) assumptions underlying these bits of conventional wisdom also change. The result is a stale policy that may no longer be effective...or possibly even dangerous.

Even the US government "standards" on password strength appear to be based on nothing more than then-current default settings on a particular operating system. Most of the "best practices" in use today are based largely on folklore or, in some cases, on severely outdated theories of password strength. These password best practices have several usability problems. Some believe that security and usability are mutually exclusive, and so security has to make things difficult. We argue that security depends on usability.

Passwords have to be strong enough to defeat cracking attempts, yet usable. This requires both an understanding of usability, and quantitative measurements of password strength. We provide an overview here and propose a solution (see [8] for more detail).

Why Do We Have Password Rules?

Users, left to their own devices, tend to choose passwords using real words. Which is understandable—users want to have a password that's easy to remember. Attackers, knowing this, use dictionaries of real words for dictionary attacks: cracking.

Password-strength rules ostensibly force the user to choose a password that's not in the attacker's dictionary. More formally, the rules attempt to prevent successful dictionary attacks by ensuring that users choose passwords with sufficient entropy to render the attack infeasible. Entropy is the measure of the probability distribution of the passwords across the keyspace—a measure of the relative randomness of each password to all the other passwords. Note that password strength rules provide no protection from brute-force attack: an exhaustive attack against the entire keyspace. The defense against a brute force attack is an immense keyspace.

Standards for Password Rules

What few standards exist are based on research that is at best inconsistent and, in most cases, appear to be pulled out of thin air. For example, NASA's password requirements claim to be in compliance with the Federal Desktop Core Configuration and are representative of these "best practices." The Core Configuration itself may contain the settings that NASA

uses, but no document within the FDCC provides any description or justification of password complexity requirements.

Here's a summary of what we could find about password rules among the various NIST and FIPS documents regarding passwords and computer security:

- Passwords shorter than 10 characters are usually considered to be weak. Passphrases shorter than 20 characters are usually considered weak (two different documents).
- Users are bad at choosing passwords; passwords should be automatically generated.
- It's difficult to measure the entropy of user-chosen passwords and there's not much data.
- Password composition is a factor in password requirements, but the specific requirements are up to the organization.
- Users should be trained on good password practices, and systems could restrict password choices based on password composition.
- Choose good passwords by using the first character of each word of a well known phrase, etc.
- When determining policies for password length and complexity, organizations should consider maximum and likely actual keyspace.
- Totally alphabetic password composition should be discouraged.

So whence the "best practices" in the NASA/FDCC requirements? It appears to come from Microsoft Windows NT Service Pack 2. NT SP-2 introduced hard-coded password strength requirements with a minimum length of six characters, and the password had to contain at least one character from the four character sets. Windows 2000 allowed for changing the settings, with an eight-character default password length. Microsoft gives no justification or citations for any of those requirements.

Additionally, the NSA [1] recommends passwords be at least 10 characters, contain at least one of each of the four character sets, and get changed every 60 days. They too provide no justification for those values.

Password Aging

> There's no there there.
>
> —Gertrude Stein, *Everybody's Autobiography*

Aging passwords—requiring users to change passwords at regular intervals—originated due to the use of hashing algorithms which were weak enough to be subject to a brute force attack. Password aging is a defense against brute force attacks, not dictionary attacks.

The NSA's Green Book details the relationship between password length and password lifetime, and includes formulae for calculating minimum password length. Note that at the time that the Green Book was written, brute-force attacks against the hash algorithms in use were considered within reach of government funded agencies.

For Windows 2000, Microsoft stated, "Where security is a concern, good values [for password lifetimes] are 30, 60, or 90 days. Where security is less important, good values are 120, 150, or 180 days." But they do not provide any definition for what "important" and "less important" are, nor how they calculated those numbers. The default password lifetime in Windows 2000 was 42 days.

None of these recommendations provide any analysis as to how much, if any, password aging reduces the risk of dictionary attacks. For any given password aging interval n, assuming some unknown attack on the passwords has equal probability of discovery at any point over n, the mean exposure time for a compromised password is $n/2$. It would seem that for any reasonable value of n, the exposure time would be unacceptable.

Passwords and Usability

> This belief of the fundamental conflict between strong computer security mechanisms and usable computer systems pervades much of modern computing. According to this belief, in order to be secure, a computer system must employ security mechanisms that are sophisticated and complex—and therefore difficult to use.
>
> —Matt Bishop, "Psychological Acceptability Revisited," *Security and Usability*

Computing professionals have long held onto the belief of an inherent tension between security and usability, that each works against the other, which has often led to a disregard of usability for the sake of securing systems. But that belief turns out to be a misconception based largely on a lack of understanding of the meaning of usability.

So what do we mean by "usability" in the context of security? Usability is often associated with perceived ease of use—the less effort required, the more usable the system. More fundamental properties of usability are [2, 3]:

- Is the user able to understand what is required of her? Can the user understand how to use the security mechanism properly, recognize when she's failed, and understand why?
- Is the user capable of using the mechanism properly?
- Does the user understand the goal of the security mechanism?
- Is the user motivated to follow the security requirements?
- Do the requirements and interface match the user's understanding of the security goals?

The study of "human factors" separates tasks into "production tasks" and "supporting tasks" (sometimes called "enabling tasks") [4]. Production tasks are the actual end goal of the user, the desired output. Supporting tasks are those that enable the user to work on the production tasks. For example, a user authenticating himself to the system enables that user to access data on the system. Accessing the data is the production task, and authentication is the supporting task. Users don't want to spend time on supporting tasks—those that have too much of an impact on production tasks affect the usability of the system and the productivity of the users.

"Ease of use" is an important property but is not completely equivalent to the "work factor"; the work factor of supporting tasks can involve not only physical time and effort, but cognitive load, the measure of the ability of people to learn [5]. The amount of mental effort a user has to expend on understanding security requirements and complying with them are all a cognitive load that affects the size of the supporting task.

In order for a security mechanism to be used properly, the user must be able to understand both how and why to use it, and be able to use it efficiently and effectively. Security depends on usability.

The Usability of Common Password Requirements

The following are common password requirements that have a negative impact on the usability of passwords:

- Rules for password complexity
- Requirements to change passwords on a periodic basis (password aging)
- Requirements not to reuse old passwords
- Prohibitions against writing down passwords

As we noted above, some of these rules were originally devised in a context that often does not apply today.

Password Complexity and Aging

Password complexity rules make the user expend time and effort to devise an acceptable password, and then memorize it. This imposes a cognitive load on the user and increases the supporting task work factor.

Password aging rules further increase the cognitive load and work factor, by forcing the user to repeat the process of devising and remembering passwords repeatedly.

The negative impact of this combination of rules has been noted in several places.

A study on password usage [6] within the FAA quantified the direct cost in staff time in changing passwords, noting that the costs were greatly magnified by the fact that users had numerous (up to 20!) passwords for different systems, all with different password rules and aging policies. Users were essentially in a steady-state of changing passwords.

This same study noted that due to the burden of remembering passwords, coupled with the impact of forgetting passwords on production tasks, users adopted numerous coping strategies, which were in turn violations of other security policies: leaving sessions logged in, sharing passwords with coworkers, writing passwords down, etc.

Even the federal government acknowledges that password changing can cause problems: "The FIPS guidelines actually acknowledge that the load on users created by frequent password changes creates its own risks, which in many contexts outweigh those created by changing a password less frequently." [4]

And here's the fun part: there is absolutely no risk justification for any of the time intervals (42 days, 3 months, 6 months, 1 year) seen in current "best practices." As far as we can tell, all of those numbers have been pulled out of thin air (or less well-lit regions).

Usability of Pro-Active Password Checking

Pro-active password checking [7] seemed like an effective approach to strong passwords at the time that it was proposed. Avoiding dictionary attacks was best solved by preventing users from entering passwords that were in the dictionary. That approach assumes, of course, that one can check against a dictionary that's at least as good as any attacker would use.

Computation power in 1992 was such that a reasonably modest dictionary of 100,000 words or so, plus common substitutions, was sufficient to deter attacks. But current computational power, combined with easy online access to comprehensive wordlists, has changed the landscape.

We made an attempt at implementing pro-active checking by doing what an attacker would do: creating the biggest dictionary we possibly could. Using 1-grams from the Google Book project. We started with a list of ~4,000,000 words, and after applying the Crack substitution algorithms, ended up with a dictionary of about 90,000,000 passwords.

Having users change their passwords while checking against the dictionary was a colossal usability failure. There were so many unacceptable words that users became frustrated trying to come up with an acceptable password, and ended up choosing randomly until one was accepted by the system.

Pro-active password checking fails usability because it's impossible for the user to understand how to comply with the rules without guessing, and ends up increasing both the work factor and cognitive load of choosing a new password.

Risks of Writing Down Passwords

The prohibition against writing down passwords is an assumed mandatory requirement [2]. So the user is forced to devise a difficult-to-remember password, and then immediately remember it, further exacerbating the cognitive load on the user [4]. Add to this the oftentimes useless feedback provided to the user while attempting to create an acceptable password.

But that risk from writing down passwords is very context dependent. Prohibition against writing passwords hails from the military, where the threat of a malicious insider (a spy) looking for written down passwords was substantial, and the liability of that risk, astronomical. That threat may be substantially lower in other contexts, where the threat of password guessing from a remote anonymous attacker is much higher.

And, as mentioned above, the burden of having to remember passwords causes users to take other measures that can impose equal or greater risks. Writing down passwords reduces the cognitive load for users, especially for passwords that get used infrequently.

Writing down passwords is also perceived as being very insecure because the passwords may get left someplace they are easily discovered. That risk can be easily mitigated with some simple rules for keeping the written password in a reasonably secure location (e.g., wallet, locked desk, etc.). Note that even the Green Book recommends that users memorize their passwords, but allows for writing down passwords as long as the written password is sufficiently secured.

In many environments, the risk of dictionary attacks against passwords greatly outweighs the risk of writing down passwords; strong passwords are more important than easily memorable passwords.

Single Sign On

The FAA study noted that many subjects had numerous passwords to remember. Reducing the number of passwords that users have to remember greatly reduces cognitive load. A single-sign-on system, where the user has to remember and use one at a given interval (once a shift, for example), has a profound effect on usability [2].

Passwords and Entropy

People often speak of password entropy as a measurement of password strength, and attempt to measure the entropy of a given password. But as stated above, entropy is the measurement of the relative randomness of all the passwords together—you can't measure the entropy of a single password.

The only way to guarantee high entropy of user-chosen passwords is to require users to enter passwords that are significantly different from other passwords. But the only way to achieve that is to reject the user's new password as being too similar to another password, which in turn provides hints about the composition of another password on the system.

Password character class rules fail to provide any guarantees of entropy because they do nothing to prevent users from choosing the same or similar passwords.

Improving the Usability and Security of Passwords at the Same Time

Here's a modest proposal to make password management more usable for users and improve the entropy of the passwords at the same time.

Provide single/common sign on to minimize the number of passwords the user must remember (reducing cognitive load) and the number of times the user has to authenticate (minimize supporting tasks).

Allow the user to write down her password, as long as it's done in a reasonably secure manner, reducing cognitive load, and reducing the need for users to adopt insecure coping strategies.

Eliminate password aging, minimizing work factor and cognitive load for devising and remembering new passwords. Only require password change when the password may have been compromised. To minimize compromise, prohibit (or at least discourage) the users from using the same password at sites out of your control.

Eliminating aging means that you need sufficient password entropy to prevent a dictionary attack. Even if you don't eliminate aging, you still need to be able to quantify the entropy in order to determine an aging interval that has acceptable risk.

So you need to implement password rules that guarantee sufficient entropy across the set of user passwords. But here's the rub: when you let users choose their own passwords, you can't devise password rules that are both usable and have enough entropy. We are publishing a paper in the near future that demonstrates this.

One answer to that dilemma is to not let users choose their passwords, but to generate passwords for them using a random algorithm. It's the easy, perhaps only, way to ensure entropy, and when done right, can be usable. At least, if the random passwords are sufficiently memorable (and typeable), they can be more usable than requiring the user to choose a complicated, difficult-to-remember password that he can't write down and has to change often.

While the cognitive load of learning the new password may be greater (and we're not sure that's true), it doesn't have to be much greater, and can be offset by allowing the user to write it down.

The above combined approach creates a "grand bargain" with the user: in return for not being able to choose her own password, the user will only have to learn the one assigned, can write it down to aid with memorization, and will never (normally) have to change it.

Reasonably Memorable Random Passwords

There is a standing assertion that random passwords are difficult to remember and therefore fundamentally unusable [3, 4]. However, these assertions turn on assumptions as to how those passwords get formed: e.g., random strings of characters. We argue that if done properly, they can be reasonably usable and memorable.

In order to randomly generate usable passwords, consider that not all users are the same; their criteria for acceptable passwords can vary:

- A short, complicated password requires less typing.
- A longer alpha-only password is easy to enter via iPhone/tablet.A very long password is easy to remember—e.g., a passphrase.

Random generation of passwords can be acceptable when the user is given a set of choices within the constraints of the password entropy requirements. Giving the user a limited set of choices also gives the user the opportunity to select a password he finds more memorable, reducing cognitive load.

To demonstrate, consider the following set of choices:

Passphrases generated from a word list	opinion parting theological infrastructure lecture vividly
Lowercase alphabetic passwords	vukizocylqhxzxiexq qgmblqmtngtiurtybj
Alpha-numeric passwords	khjd2gjact31koo7 ntrv5xbrvdbt6d05
Mixed-case alphabetic passwords	ywcgyRwIdUbBsL zmbLwdAFvQuIPQ
Random passwords	im&c<Z+I)<t^ XvG[9Hm8klpN

Our experience with this system found that the passphrases are reasonably easy to remember.

Generating memorable random passphrases requires drawing from a dictionary of words that are already well familiar to the user. The average English-speaking adult vocabulary is 20,000–50,000 words, but that list includes words the user will recognize but not know well enough to spell or remember. Using a dictionary of the 10,000 (or fewer) most frequent words seems to provide passphrases that are sufficiently memorable to the user.

Conclusion

Password rules shouldn't be used unless they're actually effective. Our proposed approach results in measurably strong passwords that we think are quite usable. But our experience to date is anecdotal; usability studies to validate our hypothesis would be a good area for future research.

References

[1] National Security Agency, "Guide to the Secure Configuration of Red Hat Enterprise Linux 5," Revision 4.2., August 26, 2011.

[2] Anne Adams and Martina Angela Sasse, "Users Are Not the Enemy," *Communications of the ACM,* vol. 42, no. 12 (December 1999), pp. 40-46, doi: 10.1145/322796.322806.

[3] J.H. Saltzer, M.D. Schroeder, "The Protection of Information in Computer Systems," *Proceedings of the IEEE,* vol. 63, no. 9 (September 1975), pp. 1278, 1308, doi: 10.1109/PROC.1975.9939.

[4] Lorrie Cranor and Simson Garfinkel, *Security and Usability* (O'Reilly Media, Inc., 2005).

[5] A.-M. Horcher, G.P. Tejay, "Building a Better Password: The Role of Cognitive Load in Information Security Training," in Proceedings of the *IEEE International Conference on Intelligence and Security Informatics* (2009), pp.113, 118.

[6] K. Allendoerfer and S. Pai, "Human Factors Considerations for Passwords and Other User Identification Techniques part 2: Field Study, Results and Analysis" (DOT/FAA/TC-06/09).

[7] M. Bishop and D. Klein, "Improving System Security Through Proactive Password Checking," *Computers and Security,* vol. 14, no. 3 (May/June 1995), pp. 233–249.

[8] A. Singer, W. Anderson,and R. Farrow, "Rethinking Password Policies" (uncut): https://www.usenix.org/publications/login/august-2013-volume-38-number-4/rethinking-password-policies-uncut.