# Choosing Passwords: Security and Human Factors

Edward F. Gehringer
Department of Electrical and Computer Engineering
Department of Computer Science
North Carolina State University
efg@ncsu.edu

## Abstract

*Password security is essential to the security of information systems. It is often recommended that passwords not be short, not be words found in a dictionary, and that they should be changed frequently. When a user has access to many accounts or systems, different passwords should be used so that no single incident will lead to the compromise of all of these accounts.*

*Unfortunately, human fallibility makes it nearly impossible to follow all of these rules simultaneously. A user with many different passwords, frequently changing, will be forced to write them down somewhere. Some systems constrain them to have a certain minimum length, or to require them to contain a combination of letters and numbers. Some systems also impose maximum lengths, and some prohibit special characters. The lack of common standards for passwords makes it difficult for a user to remember which password is used for which system. To make matters worse, systems frequently revoke a user's access after a password has been incorrectly entered as few as three times.*

*What is needed, then, is an analysis of passwords that takes both human factors and security into account. We must recognize that what really matters is the security of the total system—offline as well as online. This paper explores the tradeoffs that need to be made to achieve maximum security in everyday use by forgetful users.*

## 1. The need for good passwords

Passwords are the first line of defense against intrusion into a computer system. Users have an ethical imperative to choose good passwords to protect the sensitive information of others, and system administrators have a responsibility to see that they do. Toward that end, many system administrators provide advice to their users on how to construct a password. The following rules [1] are typical:

"A good password—

has both upper and lower case letters,
has digits and/or punctuation characters as well as letters,
is easy to remember, so it does not have to be written down,
is seven or eight characters long,
can be typed quickly, so someone else cannot look over your shoulder."

However, this was in 1991, when brute-force attacks on passwords were not as common as they are today. And the advice to use mixed-case passwords is useless for Novell and VMS systems, where passwords are case insensitive [2].

More recent advice is typified by this page from Information Services at the University of Kent at Canterbury [3]:

"When choosing a password, it [sic] really should

- Not contain words found in a dictionary. Crackers have access to very large on-line dictionaries (with more than 100,000 words), in a number of languages!
- Not be a name of a friend, relative film star or even a person in a book.
- Not be less than 8 characters in length.
- Not be a number.
- Not contain a space."

Other common recommendations include choosing a password with mixed upper-case and lower-case letters, or mixed letters and digits.

While it is obvious that you don't want to choose a password that an intruder could easily guess, many users don't appreciate the potency of the arsenal an attacker might deploy against them. In the 1980s, it was common to recommend polysyllabic dictionary words as passwords, but that is no longer prudent. Dictionaries of 60,000 words were compiled as long ago as 1990, and more recent schemes test permutations of the words, including substituting special characters for letters (e.g., "$" for "s", or "0" for "o"), or capitalizing one or two non-initial characters [4]. A cracker who gets access to a Unix password file, for example, can try these techniques at his leisure. There are, of course, more modern techniques than Unix's, but even some of these are subject to offline attacks (including the widely used Kerberos IV [5]). Few first-time users will know how strong a system's security is when they select their passwords, and hence they should choose a strong password. Thus it is recommended that passwords be at least 8 characters long and not be words found in a dictionary, or any obvious permutation of them.

But a password must also be easy to memorize. It's inconvenient to have to pull out a piece of paper every time one logs in, and, in any event, it is also a security risk, because the paper might be lost, or read by a bystander. A common suggestion [2, 3, 6, 7] is to use the initial letters of words in a phrase that you can remember, e.g., "O, say, can you see by the dawn's early light" yields "Oscysbtdel".

## 2. The problem of multiple accounts

If everyone had just a single account, the catch-phrase acronym approach would come close to solving the problem—at least if the passwords were ≥ 8 characters and (unlike the one above) weren't derived from the lyrics of well known proverbs or songs, which might someday be compiled into a hacker dictionary. But nowadays, average people have passwords for one or more computer accounts and dozens of Web sites. Systems tend to have different rules for constructing passwords, some of them quite arbitrary.

- Unix systems permit passwords of unlimited length to be typed, but only the first eight characters are significant [2], which converts some seemingly secure passwords into insecure ones, e.g., "Carolina71Duke59" becomes simply "Carolina".
- TIAA-CREF [8], the leading provider of retirement plans for college faculty, requires passwords to be between 4 and 7 characters.
- The University of Colorado Information Technology Service [9] prohibits passwords that "[c]ontain a colon (:), a pound sign (#), an exclamation mark (!), a single quote ('), a space or a tab," or "[h]ave more than 3 repeated characters — thus a password with the string 'aaa' anywhere in it would be rejected."

Even the catch-phrase acronym would be rejected by certain sites, which require that all passwords contain at least one special character.

Compounding the difficulty is a similarly inconsistent set of rules for constructing usernames, with sites having different minimum and maximum lengths, and some allowing only alphanumerics, or disallowing certain special characters. These rules combine to make it virtually impossible for users to remember their login/password combinations without writing them down, either on paper or in a file. Indeed, Dhamija and Perrig [10] found that more than a quarter of users failed to recall not only their passwords, but also their usernames.

Thus, it is impossible to use the same username, password, or username/password combination for all sites to which one has access. This helps diminish a serious security risk, since some sites may store passwords in plaintext accessible to administrators. For example, I was distressed to discover earlier this year that a help-line support person at my long-distance provider was able to quote me a password that I couldn't recall. If that had been a password I used on other sites, he could have read my credit-card statements (they had my credit-card number), transferred money between my bank accounts, or sold my mutual funds. Indeed, the interloper might easily have acquired enough information about me to steal my identity, something that happened to 700,000 consumers in 2001 [11].

On the other hand, writing down a list of one's accounts and passwords may even be a greater security risk, because anyone who comes into possession of such a list will automatically be able to access the victim's accounts, without having to guess which institutions hold them. This risk is widespread, as evidenced by one study [12] that found that 50% of users wrote their passwords down. We will return to some proposed solutions later, but first, let us consider the implications of two other schemes for protecting passwords: the requirement that they be changed at regular intervals, and "locking out" accounts after a certain number of unsuccessful login attempts.

## 3. Lifetime limits and lockouts

System administrators frequently require users to change their passwords at regular intervals. This is intended to increase security by denying an intruder long-term access to an account. For example, Federal Information Processing Standards Publication 112 [13] specifies that passwords on computers operated by any agency of the U. S. Government "shall have the shortest practical lifetime, selected by the Security Officer in conjunction with the Systems Manager, which provides the desired level of protection at the least possible cost …" In no event may the lifetime be greater than one year.

Adams and Sasse [12] identify several problems with lifetime limits. When required to change passwords frequently, users are eventually forced into using less memorable passwords, or get confused as to which password they are actually using, either of which increases the tendency to write passwords down. Or, in an effort to remember, they choose progressively simpler passwords, which are less secure. This tends to make users—few of whom appreciate security risks anyway—more cynical about security in general. Adams and Sasse conclude, "Although change regimes are employed to reduce the impact of an undetected security breach, our findings suggest they reduce the overall password security in an organization."

Lockouts are a technique to prevent password discovery through brute-force techniques. After a certain number of unsuccessful attempts, the system will lock up an account and deny further access, even if the correct password is subsequently entered. Intervention by the system administrator is needed to re-enable the account. Three unsuccessful guesses is a typical limit [5]. This is quite reasonable in the case of a user with a single account and no password-lifetime limit. But in an environment where a user has numerous accounts with different passwords, it is a powerful incentive to write down a list of those passwords rather than guess at them when trying to log in.

Three guesses is probably an unnecessarily low threshold anyway. Zviran and Haga [14] performed an experiment where users chose "cognitive" passwords—passwords based on personal facts, interest, and opinions that are likely to be recalled by a user. Most such passwords are very insecure by today's standards. Yet these passwords were guessable only 29% of the time by the "significant others" in the users' lives—even though the significant others were told what fact the password was based on (e.g., the name of the elementary school from which the user graduated). A clear implication of this study is that an intruder would have very little chance of guessing a user's password in the first few guesses.

An alternative technique to foil password-guessing is to introduce a delay of a few seconds between attempts. This alternative seems to be unlikely to induce users to write passwords down, since a forgetful user is likely to need a few seconds' think time between attempts. Rubin [5] suggests combining this technique with lockouts for greater security. But if lockouts are to be used, the threshold could be set high enough (say, one to two dozen attempts) so that it does not create an incentive to write down passwords.

Taken together, multiple accounts with lifetime limits and lockouts could produce the worst password security of all. Without reusing the same password across multiple sites, a user would have no way of remembering which password applied to which site at which time. Using the same password for all sites would help a little, but since users may have accounts at Web sites they visit infrequently, they would still need to remember a succession of "old" passwords. This, coupled with the inability to try more than two or three passwords before being locked out, would very nearly guarantee that users would write down all their passwords and keep the lists readily accessible. Intruders would know this, and the theft of such lists would become a major security problem.

## 4. Wallets

The proliferation of passwords has given rise to a host of software applications designed to help users manage their accounts. Collectively called "wallets," they come in two different varieties. The first is a username/password repository, which is essentially an encrypted file kept on your computer that holds information you need to log into your various accounts. The most prominent of these is Darn! Passwords! [15]. It has a password generator that can make up passwords for various applications, and allows you to drag your passwords into the application or Web site that you are using. It allows you to remember one password instead of many. Similar applications are Password Safe [16] and Q*Wallet [17], both for Windows. Selznick PasswordWallet [18] provides similar functionality on the Macintosh and Palm OS. Apparently no similar product exists for Unix or Linux.

While undoubtedly a convenience for users, these programs do not completely solve the problem of achieving password security. They are only as secure as the password the user chooses for the wallet and the physical security of the user's computer. Network accounts and Web sites may have rules that require strong passwords, but a user can circumvent these by using a weak password for the password wallet. Though such passwords are not normally passed over a network, they may be susceptible to virus attacks that would expose them to an interloper. Anyone who has physical access to a computer running a wallet program would have access to all the passwords, so users would have to remember to lock their computer or quit the application when walking away from their desks [19].

The other style of wallet program holds not only passwords, but other information that a user might need in accessing a Web site, and aims to facilitate moving from site to site without re-entering information. The most prominent application in this category is Microsoft's Passport [20], which is targeted at consumer-oriented shopping sites. The Passport server maintains personal information about the customer, including such items as credit-card numbers and shoe size, and passes this information on, with permission, to sites that the consumer visits. The client begins by connecting to the merchant's site. When the customer needs to authenticate, the merchant server redirects the user to a Passport server. The user logs in at the Passport server, which then redirects the user back to the end server [5].

Passport does not work with all sites, only with participating merchant sites, so the user still needs to keep track of usernames and passwords to other sites. There is also the possibility that an intruder could observe the network between the client's browser and the merchant's Web server and impersonate the Passport server in order to read the user's Passport password. This is possible because Microsoft, in an effort to make the service as general as possible, did not protect the redirection at the beginning of a Passport session by SSL [5]. This is one of several risks in its protocol [21].

## 5. One possible approach

One approach to devising passwords that are different for each site, secure, and memorable, is suggested by Craig Busse [22]. He suggests using a password embedding an anagram of the site name:

> "E.g., take the first 2 letters of the URL (or user ID or hostname or whatever you are logging onto), in reverse order as the first two letters of the password and your own initials, again in reverse order, as the last two letters of the password. Insert digits 01234 ... as needed in the middle to satisfy any length requirements."

He suggests that readers choose their own strategy similar to, but not identical to that. This seems to be more secure than using the same password on all sites, though it would easily be crackable by an attacker that managed to gather enough of a user's passwords to spot the pattern. It also does not deal with change requirements, but those could be handled by varying the filler text to, e.g., encode the month that the password was changed. Obviously, this sacrifices much of the value of changing passwords in order to combat undetected security breaches, but at least it does not reduce security by tempting users to write their passwords down.

## 6. Recommendations

Both users and administrators have ethical responsibilities to maintain good password security. Users have obligations not only to themselves, but also to their fellow users, whose security may be compromised if an intruder gains access to the computer system. They should use passwords that do not appear in the dictionary, that are not too short, that do not encode any of their personal or account attributes, and that follow any other relevant rules from Section 1. They should consider encoding site names, dates, and other information in passwords, as outlined in Section 5. Administrators have the primary responsibility for keeping their system and network secure. They should make sure users are aware of the attributes of a good password. They should refrain from imposing unrealistic restrictions that invite circumvention, as these leave the system vulnerable and promote cynicism about security.

Specifically, systems should avoid restrictions on passwords that are motivated by programming convenience rather than security. Thus, the maximum password length should be at least thirty characters, or should be unconstrained altogether. No characters should be prohibited from appearing in passwords; full Unicode should be allowed if the system supports it. Systems should rarely impose a maximum lifetime on passwords. One exception might be a security-critical system in daily use by most of its users; if change regimes are imposed only on the one system that its users use most frequently, they will be more manageable than if users are required to change passwords on multiple systems at varying intervals. Similarly, lockouts after failed login attempts should be used sparingly if at all. Enforced delays after unsuccessful attempts serve the same purpose, and are not likely to induce users to write down passwords.

Rules on what kinds of characters a password *should* contain are more justifiable than rules on what characters it should not contain. But, rather than mandating the use of uppercase, digits, or special characters, it is probably better for a system to evaluate each new password against some metric and reject those that are too weak, telling the user why they were rejected. This avoids proscribing, say, a 20-character phrase with idiosyncratic word breaks just because it does not contain a special character.

## 7. Summary

Maintaining password security involves striking a delicate balance between having enough rules to maintain good security and not having so many that users will take evasive action that compromises security. To date, most of the emphasis in the literature has been on having strong enough rules. Only two articles the author encountered [12, 19] focused on the pitfalls of having too-stringent rules. True security, however, is an attribute of the entire human-computer environment, not just what is stored digitally. Future work in this area should not leave the human out of the equation.

## 8. Acknowledgments

The author acknowledges the advice of Ruth Day, Duke University Department of Psychology, for her suggestions on what sources to consult.

## 9. References

[1] Simson Garfinkel and Gene Spafford, *Practical UNIX Security*, Sebastopol, CA: O'Reilly & Associates, Inc., 1991, p. 35.

[2] Centre Européen pour Recherche Nucléaire, "Passwords," http://wwwinfo.cern.ch/pdp/ose/security/cern/service/passwords.html

[3] University of Kent at Canterbury Computing Service, "Choosing a password," http://www.ukc.ac.uk/informationservices/computing/windows2000/publicpc/passwords/choose.html

[4] William Stallings, *Network Security Essentials: Applications and Standards*, Prentice-Hall, 2000, Chapter 9.

[5] Aviel D. Rubin, *White-Hat Security Arsenal: Tackling the Threats*, Addison-Wesley, 2001.

[6] University of Pennsylvania Computing, "Choosing passwords," http://www.upenn.edu/computing/security-privacy/passwords.html

[7] University of Illinois at Chicago, Academic Computing and Communications Center, "Choosing a safe password," http://www.uic.edu/depts/accc/accts/password.html

[8] TIAA-CREF, "Secure access to the TIAA-CREF Web Center," https://ais2.tiaa-cref.org/cgi-bin/WebObjects.exe/IndvGate?faCmd=PACCreate

[9] University of Colorado, Information Technology Services, "ITS guidelines for choosing passwords," http://www.colorado.edu/ITS/CR/UserInfo/passwd.html

[10] Rachna Dhamija and Adrian Perrig, "Déjà Vu: A user study using images for authentication," *Usenix Security Symposium*, 2000. Available at http://paris.cs.berkeley.edu/~perrig/projects/usenix2000/usenix.pdf.

[11] Identity Theft Resource Center, http://www.idtheftcenter.org

[12] Anne Adams and Martina Angela Sasse, "Users are not the enemy," *Communications of the ACM* 42:12, December 1999, pp. 40–46.

[13] Federal Information Processing Standards Publication 112: Password Usage, http://www.itl.nist.gov/fipspubs/fip112.htm

[14] Moshe Zviran and William J. Haga, "User authentication by cognitive passwords: An empirical assessment," *Proc. IEEE Jerusalem Conference on Information Technology 1990: Next Decade in Information Technology*, pp. 137–144.

[15] "Darn! Reminder Software!" EmmaSoft, 2002, http://www.odarn.com

[16] Password Safe 1.7.1, Counterpane Labs, http://www.counterpane.com/passsafe.html

[17] Q*Wallet, http://qwallet.com

[18] The PasswordWallet™ Family of Products, Selznick Scientific Software, LLC, http://www.selznick.com/products/passwordwallet

[19] Wayne Rash, "Password chaos threatens e-commerce," ZDNet Tech Update, February 19, 2002, http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2847895,00.html

[20] Microsoft.NET Passport, http://www.passport.com

[21] David P. Kormann and Aviel D. Rubin, "Risks of the Passport single signon protocol," *Computer Networks* 33:51–58, 2000. Available at http://www.avirubin.com/passport.html

[22] Craig Busse, "Talk back" posting in response to reference 19 at http://forums.zdnet.com/group/zd.Tech.Update/it/itupdatetb.tpt/@thread@2008@F@1@D-,D@ALL/@article@2147?EXP=ALL&VWM=hr&ROS=1