Password Generator Project

> *“Choosing a hard-to-guess, but easy-to-remember password is important!”—Kevin Mitnick*

Password Vulnerabilities

In March 2013, the popular online note-taking service, Evernote, issued aSecurity Notice alerting users of a “service wide password reset” that they were enfocing as a result of a “coordinated attempt to access secure areas of the Evernote Service.”

Fortunately, no personal information or data was breached and that by issuing a required password reset, Evernote was merely taking proactive steps in an abundance of caution.

However, such widespread password leaks and security breaches are becoming all too common as online services play a larger and larger role in our digital lives. Several times every year, the news reports of yet another hack of a popular site or a leak of passwords and other sensitive user data.

In its announcement, Evernote offered its users advice to follow that could help to ensure their data on*any* site.

> Advice from an Evernote Security Notice (March 2, 2013):
> - *Avoid using simple passwords based on dictionary words*
> - *Never use the same password on multiple sites or services*
> - *Never click on ‘reset password’ requests in e-mails—instead go directly to the service*

The first two bullet points on this list relate to the two most likely mistakes that users make when choosing a password. They relate to the dilemma of trying to balance the security of complex passwords with the convenience of having a single, easy-to-remember password.

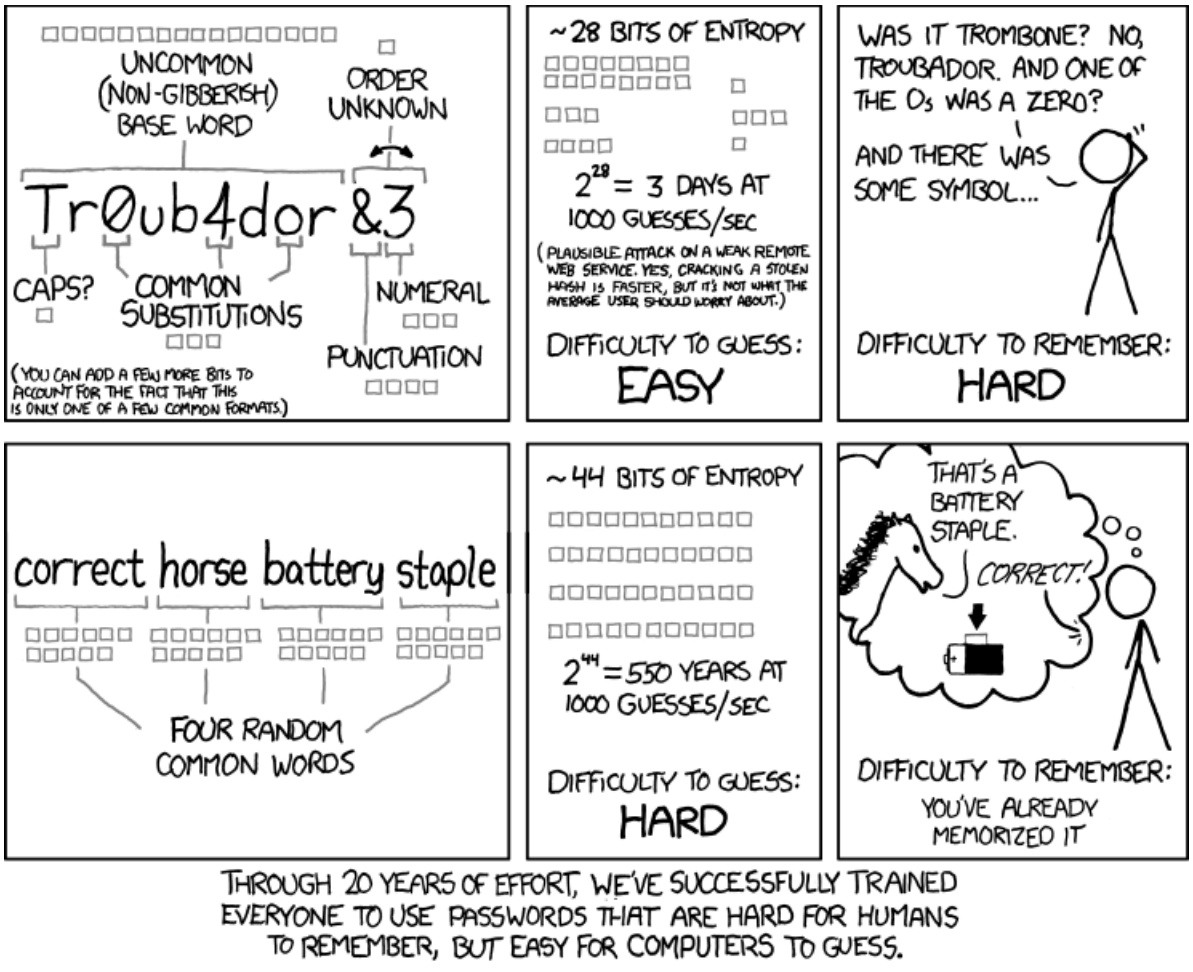
Unfortunately, despite their concerns over privacy, people often choose convenience over security. Let’s take a closer look at two ways that users often leave themselves vulnerable to attack.

Easy to Remember, Hard to Guess

> *"Any password that can be easily remembered is vulnerable to a dictionary attack.”— Bruce Schneier*

The goal of any password should be to choose something that is difficult to guess, especially when some automated systems are capable of making millions of guesses per second. People use a number of techniques to increase the complexity of their passwords, such as mixing upper- and lowercase letters, substituting digits and punctuation for letters, appending extra characters or numbers, etc. Unfortunately, while these efforts might increase the effort required to guess them, it also increases the difficulty of remembering them.

A popular xkcd comic by Randall Munroe addressed the issue of how difficult it is to crack a password vs. how easy it is to remember by attempting to measure password strength in terms of “bits of entropy.”



Unfortunately, as security expert Bruce Schneier notes in his article, "Choosing Secure Passwords," despite Munroe's logic, his suggested solution is actually quite vulnerable to attack due to its reliance upon common *dictionary words* that are easy to guess by brute force.

So, just how does one create a password that is strong but memorable? What steps could you take to create a secure password that is convenient for you to use? Discuss some ideas with your partner and be prepared to share your suggestions with the class.

Single Point of Failure

Another common mistake that people frequently make with their password security was also addressed by Evernote's advice to their users—"Never use the same password on multiple sites or services."

There are many reasons why users might reuse the same password for every site or service they visit, with the most obvious being that it is simply easier to remember *one* password than it is to remember *many* different passwords. It is common sense. But it is also highly insecure.

The problem is that a password is meant to be a secret credential that you use to identify yourself to someone else. This method of authentication relies on the basic assumption that there is a one-to-one relationship between knowing the password and having the right to access an account. That is, in theory, only the owner of an account can provide the secret piece of information (i.e., the password) that can confirm the individual's identity. However, if any other, third party should ever possess that secret information, then the initial assumption fails as knowing the password is no longer guarateed to be limited to the account owner.

The real problem lies in the fact that in order for someone else (e.g., Facebook, Amazon, your bank) to be able to authenticate you, they need to know what the secret piece of information is that only you can tell them—which *itself* violates the first

assumption and invalidates your use of that password with any other party.

For example, imagine you sign up with Facebook and set your password to be `fuzzybunny123`. Now *you* know your password. And *Facebook* knows your password. So far, so good. Whenever you visit Facebook, you can prove to the site that you are you by identifying yourself with the `fuzzybunny123` password that only you and Facebook know. However, you also sign up for an account with some other social sharing site, like *InstaChatOmatic*, and you use the same `fuzzybunny123` password that you use with Facebook. Now, three different parties all know the same information that, in theory, only *you* should know. A malicious employee, rogue software, or database leak at InstaChatOmatic can result in an unknown party attempting to access your Facebook account using your InstaChatOmatic password. And since the passwords are the same, Facebook will authenticate the *intruder* as if it is *you* because they will have provided that information that supposedly only*you* should have known.

In short, by reusing the same password with more than one service, you've undermined the security of your password at both sites—all in the name of making it easier for *you* to remember. This is flawed thinking. Passwords are meant for security and the strength of that security should be prioritized above anything else that undermines it.

Remember Algorithms, Not Passwords

In his article, "Passwords Are Not Broken, but How We Choose them Sure Is," Schneier suggests what he calls the "Schneier Scheme":

> *"My advice is to take a sentence and turn it into a password. Something like 'This little piggy went to market' might become* `tlpWENT2m` *. That nine-character password won't be in anyone's dictionary. Of course, don't use this one, because I've written about it. Choose your own sentence—something personal."—Bruce Schneier*

The important thing to note about this approach is that rather than trying to remember an obscure collection of odd and difficult to guess letters, digits, and punctuation, one needs only to remember a personalized phrase or other mnemonic that will remind you how to easily reconstruct the password.

And if you customize the key phrase to match the site or service, a single, simple set of rules can be created that will allow you to easily reconstruct the password anytime you visit the site. For example, consider the following phrase/password scheme:

- *"Facebook is where I post to my friends."*… `FB8pstBFFS`
- *"Gmail is where I read my mail."*… `GM5rdMAIL`
- *"Twitter is where I follow my friends."*… `TW7fllwBFFS`
- *"YouTube is where I watch videos."*… `YT7wtchVIDEOS`

The single algorithm that you would need to memorize for this scheme would be something like the following:

```
1) Abbreviate the site into a 2-letter phrase.
2) Capitalize the site abbreviation.
3) Type the site abbreviation.
4) Type the number of letters in the site name.
5) Identify the verb that describes how you use the site.
6) Remove all vowels from the verb.
7) Type the vowel-less verb in lowercase letters.
8) Identify the subject or type of content for the site.
9) Capitalize the subject or type of content.
10) Type the capitalized subject or type of content.
```

What are the advantages to this particular algorithm? Are there any problems or weaknesses to this algorithm? What other algorithms can you think of that might work better? Be prepared to discuss your ideas with the class. (Of course, if you have a *really* clever idea, you might want to save that one for yourself and only discuss your rejected ideas.)

Assignment

> ***Design an algorithm that can generate a custom, reproducible password that is uniquely different for each website.***

Working in pairs, your task is to design and construct a standardized strategy for generating unique passwords for different sites that can later be regenerated by reapplying the same algorithm. Your solution should address the following concepts:

- The algorithm should generate different passwords for different sites.
- The password for any site should be reproducible simply by following the algorithm.
- The algorithm should be easy to remember and apply.
- The password should be complex and difficult to guess.
- The general algorithm should not be easily deduced from the password.

Once you've designed your solution, write out each step of your password-generating algorithm in some form of *pseudocode*. No specific format is required for your algorithm, but your pseudocode should be clear enough and detailed enough that anyone who is not familiar with how your algorithm is supposed to work can still follow along and apply its steps in generating a valid password.

Submission

Your submission will be in the form of a written algorithm (i.e., pseudocode) that explicitly states each of the discrete steps and decisions that must be made in generating a valid password. Also, you must provide at least five examples of passwords that your algorithm would generate for five different sites. One of those examples must be thoroughly annotated, showing how each step of the algorithm contributes to the final password.

Your solution and examples should demonstrate the following properties:

- Clear and readable
- Cleanly formatted
- Appropriate use of sequencing, selection, and/or iteration
- Well-documented examples

Learning Goals

Over the course of this module and this project, you will learn to:

- identify and examine a number of common features of algorithms, including sequencing, selection, and repetition
- write pseudocode to describe each step of an algorithm with clarity and precision
- construct trace tables documenting the result of each step of an algorithm
- compare the differences between different types of common algorithms
- analyze the need for artificial programming languages
- examine strategies for approaching large-scale problems
- identify factors that allow solutions to scale efficiently

- encode and decode messages using common cryptographic techniques
- examine a number of common threats to cybersecurity, including distributed denial of service attacks (DDoS), phishing, viruses, and social engineering
- examine the implications of Moore's Law on the research and development of new and existing technologies

Rubric

Content Area	Performance Quality			
Readability	Algorithm is typed, organized, and nicely formatted for easy use.	Algorithm is organized and nicely formatted for easy use, but is not typed. **—OR—** Algorithm is typed, but the formatting and organization makes it somewhat difficult to use.	Algorithm has formatting and organization that makes it somewhat difficult to use **AND** is not typed. **—OR—** Algorithm may be typed, but the formatting and organization makes it extremely difficult to use.	Not enough criteria are met in order to award any credit.
Flow	The algorithm incorporates the appropriate use of all three types of programming structure: sequencing, selection, and iteration.	The algorithm incorporates the appropriate use of only two types of programming structure: sequencing, selection, and iteration.	The algorithm incorporates the appropriate use of only one type of programming structure: sequencing, selection, and iteration.	Not enough criteria are met in order to award any credit.
Correctness	The algorithm generates a unique and reproducible password for all sites.	The algorithm generates a reproducible password for all sites, however, some may not be unique. **—OR—** The algorithm generates a unique and reproducible password for most sites. **—OR—** The algorithm generates a unique password for all sites, however, it is not reproducible.	The algorithm generates a password for all sites, however, some may not be unique or reproducible. **—OR—** The algorithm generates a unique and reproducible password for only a few sites.	Not enough criteria are met in order to award any credit.
Effectiveness	The algorithm cannot be easily deduced from just the password and the name of the site.	A few parts of the algorithm can be easily deduced from just the password and the name of the site.	Most parts of the algorithm can be easily deduced from just the password and the name of the site.	Not enough criteria are met in order to award any credit.
Examples	There are five sample passwords generated correctly based on the algorithm.	There are four sample passwords generated correctly based on the algorithm.	There are three or fewer sample passwords generated correctly based on the algorithm.	Not enough criteria are met in order to award any credit.

Documented Case	There is one annotated example documented at all steps of the process. **—AND—** It is well formatted and organized and easy to follow.	There is one annotated example documented at most steps of the process **AND** It is well formatted and organized and easy to follow. **—OR—** There is one annotated example documented at all steps of the process, but the organization and formatting makes it difficult to follow.	There is one annotated example documented at some steps of the process **AND** It is well formatted and organized and easy to follow. **—OR—** There is one annotated example documented at all steps of the process, but the organization and formatting makes it extremely difficult to follow. **—OR—** There is one annotated example documented at most steps of the process, but the organization and formatting make it difficult to follow.	Not enough criteria are met in order to award any credit.