Making Random Letter Passwords Memorable

Making Random Letter Passwords Memorable

Arnold G. Reinhold
Cambridge, Massachusetts, USA

August 28, 2011

Abstract

A method is presented that accepts a random string of up to 10 letters and uses a look-up table to produce a mnemonic English sentence having those letters as the initial letter of each word. This method offers more predictable security than asking users who wish to create a strong password to think of a sentence and use the initials of each word in that sentence as the password.

Introduction

As personal computers become more powerful, attacks on password management systems are becoming difficult to prevent. Many authentication systems protect passwords by only storing a cryptographic hash of each password. However, if an attacker gains access to stored password hashes, they can attempt to crack the hashes using dictionaries of common passwords or brute force searches of all character combinations. The availability of high performance general-purpose graphics processors (GPGPUs) that can be programmed to carry out hash attacks exacerbates the problem. (Davis 2011) Cybercriminals have assembled large networks of computers they control (botnets) and can use CPUs and GPUs on the compromised machines to attack password hashes they have collected. Passwords used to generate cryptographic keys, such as those used for disk encryption or to protect wireless networks, have similar vulnerabilities.

In response to these threats, users are often encouraged (or coerced) to employ stronger passwords. Common ways of doing this include requiring a minimum length and a mix of upper and lower case letters, number and special characters. The latter approach can have mixed results. Users often follow predictable strategies to meet those requirements, modifying their passwords in minimal ways that have only modest impact on an attacker's search difficulty.

Another common strategy suggests users think of a phrase that is memorable and use the initial letters of each word in that phrase to form their password, perhaps substituting words or letters with numbers or symbols, 2Bor~2B? for example .However this approach depends on the user being sufficiently clever and creative. Many users will choose common phrases, such as lyrics from popular songs, and use predictable letter and word substitutions, such as "$" for "S", "1" for "L" and "3" for "E". Also the distribution of initial letters of words in English and other natural languages is far from uniform, giving attackers an additional advantage.

A more rigorously secure method is to offer users a password made up of characters chosen completely at random. Selecting uniformly distributed random characters offers the maximum possible entropy for a given password length and character set. However, this approach is not widely employed because of concerns that users find such passwords too difficult to remember.

The sentence generator approach

This paper proposes a different approach that offers the best features of the last two methods. A fully random password is created first and that password is then used to generate a mnemonic sentence – one where the initial letters of each word form the password. Since the random password is generated first, the selected sentence cannot diminish security, as long as it is kept secret.

The sentence is generated from the password using s look up table, Table 1, which consists of 10 columns and 26 rows. Every column has 26 English words in alphabetical order, each starting with a different letter of the alphabet (a-z). The one exception is the “x” row where words starting with “ex” are used. Each column contains one grammatical form. The columns are arranged to produce a proper English sentence regardless of which row each word comes from. The column pattern from left to right is: proper name, adjective, noun, adverb, verb, adjective, noun, gerund, adjective, noun.

Give any string of 10 letters from the English alphabet, Table 1 can generate an English sentence consisting of ten words whose initials are those letters (with words that start with “ex” standing for “x”). Thus any random password of 10 letters produces a unique mnemonic sentence for that password. For example the password vmyhvxklke generates the sentence:

Vivian's merry yankees hopelessly view excellent kings leaving keen energy.

While not all sentences generated in this way are immediately meaningful, they can be easier to remember than the password itself. Simple techniques, such as visualizing the scene suggested by the sentence, can further aid memorization.

Table 1. Sentence generating matrix

--	Name	adj	noun	adv	verb	adj	noun	gerund	adj	noun
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
A	Arnold's	amazing	artists	always	arrest	angry	ants	arousing	awful	admiration
B	Bob's	big	brothers	boldly	batter	bossy	boys	bringing	boastful	bliss
C	Charlie's	cuddly	cats	craftily	cover	crazy	crooks	causing	cold	comfort
D	Dona's	deadly	ducks	deftly	drop	dumb	doctors	defying	dumb	delight
E	Ed's	empty	editors	easily	engage	eager	eels	enjoying	easy	energy
F	Frank's	fine	frogs	foolishly	fight	fat	foxes	finding	fast	fame
G	Gloria's	golden	goats	gaily	grab	green	goons	gaining	glorious	growth
H	Hana's	hot	hippos	hopelessly	hold	heavy	horses	helping	happy	health
I	Ivy's	interesting	infants	intensely	inject	incompetent	idiots	insulting	intense	interest
J	Jane's	jolly	judges	joyously	join	jealous	jokers	joining	juvenile	joy
K	Ken's	kissable	kittens	kindly	keep	kinky	kings	killing	keen	karma
L	Lucy's	lonely	llamas	laughingly	lash	lowly	librarians	leaving	lurid	love

M	Mary's	merry	mermaids	morosely	mangle	mad	monsters	making	messy	music
N	Nancy's	nice	nuns	noisily	nab	naughty	nerds	noting	neglected	nothingness
O	Olga's	old	owls	often	ogle	oily	orcs	owning	open	obsession
P	Pete's	pink	peacocks	playfully	pester	poor	pigs	packing	proud	power
Q	Quincy's	quiet	quails	quickly	query	quaking	queens	questioning	queer	quality
R	Randy's	red	rodents	regretfully	ruin	rude	robbers	rejecting	redolent	refreshment
S	Sue's	smooth	snails	swiftly	slay	snarky	slugs	seeking	simple	success
T	Tom's	tiny	tigers	timidly	tackle	tired	thugs	testing	tenuous	truth
U	Uri's	urban	umpires	urgently	upset	ugly	uncles	urging	useless	unity
V	Vivian's	vivacious	vampires	vividly	view	vicious	vandals	viewing	velvet	victory
W	Walt's	wild	wolves	willingly	wrestle	wimpy	wardens	wishing	witty	wisdom
X	Xavier's	eXotic	eXecutives	eXcitedly	eXpel	eXcellent	eXperts	eXtracting	eXtreme	eXcess
Y	Yolanda's	yelping	yankees	yearningly	yank	yellow	youths	yielding	yummy	yogurt
Z	Zed's	zigzagging	zebras	zealously	zone	zany	zombies	zooming	zesty	zeros

Strength of all-letter passwords

A 10 letter random password has 10 x $\log_2$ (26) = 47.0 bits of entropy. For higher security, two sentences can be generated. If both are 10 letters long, the resulting password will have 94 bits of entropy, well exceeding NIST 800-63 guidelines for cryptographic strength (80 bits).

The table can be easily adapted to generate shorter sentences. Thus a 9 letter password would simply omit column 9, Vivian's merry yankees hopelessly view excellent kings leaving energy. An 8 letter password could omit columns 8 and 9 while making the word from column 8 possessive: Vivian's merry yankees hopelessly view excellent kings' energy . For 7 letters, omit the last three columns, and so on.

Thus a 17 letter password, offering 80 bit security, could be represented by a 10 word sentence and a 7 word sentence, or a 9 and an 8 word sentence.

Random passwords consisting of only English letters have less entropy per character than random passwords selected from a larger character set, but additional letters can be added to make up the difference. For example, a random 10 English letter password has a bit more entropy than a 7 character random password selected from all printable 7-bit ASCII characters (95 possibilities), which has 45.9 bits of entropy. To match the 65.7 bit entropy of a 10 character all-printable-ASCII password such as U{l>gPzH:Z requires an English letter password with 14 letters, which is longer, but arguably more memorable, at least when used with the method proposed here. Note that on many mobile devices, such as the Apple iPhone, it is more difficult to

type a password randomly selected from all printable ASCII characters because multiple shifts are needed to access different groups of characters.

Other security impacts

The sentence generating approach table has few security limitations. Of course, the sentence generated must be afforded the same level of security protection as the password itself. And asking users to submit their password over the Internet to get their sentence has obvious security risks. It is better to display the sentence at the same time the password is generated or perform the table lookup locally.

The choice of words has no security implication as long as there is one word for each letter in the alphabet in every column. The words in Table 1 were selected to maximize the likelihood of a somewhat meaningful sentence, while minimizing the likelihood of a sentence with sexually suggestive or scatological meaning. While some might find an X-rated sentence easier to remember, others might find such sentences offensive and organizations might be reluctant to employ such tables to avoid creating a hostile work environment.

Tables for other languages and alphabets are feasible. Different tables could be created for variety, particularly when more than one sentence is needed to meet strength objectives. Other possibilities for password mnemonics include random poems, songs, haiku, limericks and similar short literary works.

Implementation

A 10 letter random password can be selected uniformly from the English alphabet using a strong random number generator, such as CryptGenRandom on Microsoft Windows systems, and /dev/random on Unix, Linux or MacOS X systems. The Python programming language has a SystemRandom class that uses either CryptGenRandom or /dev/random, depending on the operating system on which it is running.

An ideal implementation would be to offer a user a randomly generated password and a mnemonic sentence when a new account is created or a password is to be changed on an existing account.

Users wishing to use this system needn’t wait until it is adopted by password management systems. Strong random passwords can be generated manually using dice, playing cards or letter tiles. (Reinhold 2000)

If numbers, upper case letters and special characters are needed to meet password composition policy, they can be added easily. Nouns can be capitalized and normal sentence punctuation added. Such steps add security only if an attacker is unaware of the method used, however they never diminish security. Net additional security can be achieved by prefixing the first and second adjectives with a random number, in which case 3.3 bits of entropy are added per digit. Such a sentence would still be meaningful to a user, for example, Vivian's 23 merry yankees hopelessly view 7 excellent kings leaving keen energy would yield the password v23myhv7xklke.

Conclusion

The difficulty of getting users to employ strong passwords is a major challenge to cyber security. The method presented here can help in that effort by giving users an easier way to remember a random password.

References

Davis 2011, Joshua Davis and Richard Boyd, *Teraflop Troubles: The Power of Graphics Processing Units May Threaten the World’s Password Security System,* Georgia Tech Research Institute Case Study, 2011

Reinhold 2000, Arnold Reinhold, Picking a strong Passphrase using Diceware, Internet Secrets, 2nd Edition, John R. Levine, Editor, Chapter 37, p. 831 IDG Books, 2000, ISBN 0-7645-3239-1, also www.diceware.com.