All.Net Analyst Report and Newsletter

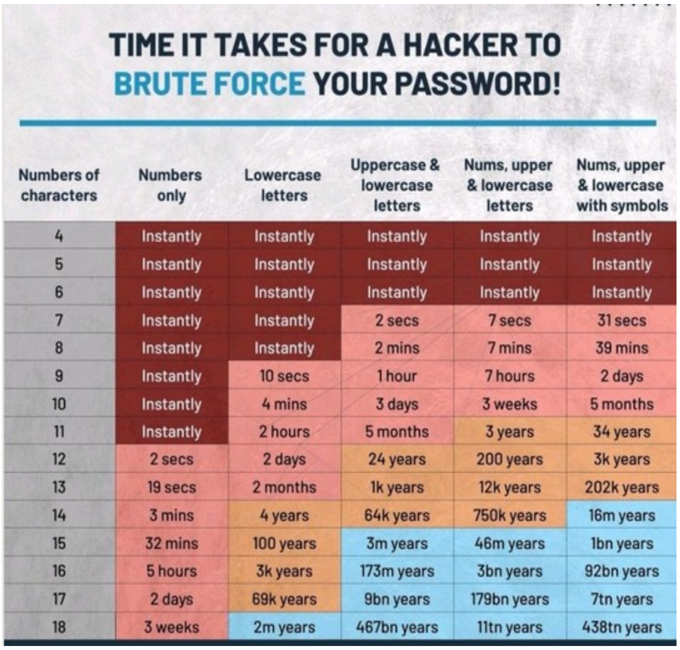
Welcome to our Analyst Report and Newsletter

How long to guess your password?

I periodically see a chart like this one that is completely misleading. It makes an assumption that the attacker has access to guess a password $52^7/2$ times per second. That's over 500 billion tries per second.

Of course guessing a real password on a real system usually takes more like 1 second, and for systems that slowly provide the password prompt in a separate faded in window and then slowly go check it, it's more like one guess per 10 seconds.

And that means that instead of taking 2 seconds for a 7 character upper/lower case password, it will take more like 500 billion seconds, or more than 16 thousand years; 160 thousand years at 10 seconds each, on average, to guess a randomly selected sequence of those symbols.



TIME IT TAKES FOR A HACKER TO BRUTE FORCE YOUR PASSWORD!

Numbers of characters	Numbers only	Lowercase letters	Uppercase & lowercase letters	Nums, upper & lowercase letters	Nums, upper & lowercase with symbols
4	Instantly	Instantly	Instantly	Instantly	Instantly
5	Instantly	Instantly	Instantly	Instantly	Instantly
6	Instantly	Instantly	Instantly	Instantly	Instantly
7	Instantly	Instantly	2 secs	7 secs	31 secs
8	Instantly	Instantly	2 mins	7 mins	39 mins
9	Instantly	10 secs	1 hour	7 hours	2 days
10	Instantly	4 mins	3 days	3 weeks	5 months
11	Instantly	2 hours	5 months	3 years	34 years
12	2 secs	2 days	24 years	200 years	3k years
13	19 secs	2 months	1k years	12k years	202k years
14	3 mins	4 years	64k years	750k years	16m years
15	32 mins	100 years	3m years	46m years	1bn years
16	5 hours	3k years	173m years	3bn years	92bn years
17	2 days	69k years	9bn years	179bn years	7tn years
18	3 weeks	2m years	467bn years	11tn years	438tn years

WRONG ANSWER!!! BAD ASSUMPTIONS

How do they get to those scary numbers?

Bad assumptions mean bad conclusions. The assumption is that the attackers have access to a "password file" containing user identities and cryptographically checksummed hashes of the passwords. In that case, if they can execute the hashing algorithm quickly enough, or pre-generate the results for a "rainbow table", they can get a valid password quickly. But to gain access to the password file means they already bypassed the security of the system enough to gain access to a protected file, which means they already have the access they could gain by guessing all the passwords… or in other words, once they get in, they can get in…

What would be reasonable?

The problem is, it's hard to remember even a 7-character randomly selected upper or lower case sequence of letters. So people tend to choose non-randomly. So let's look at something easier. Suppose we randomly generate passwords for users consisting of 3 1-syllable English words. According to Google there are 9268 of these, so a random sequence of 3 leads to 9268^3 total passwords, almost 800 billion of them. Better than 7 random upper/lower case letters, easy to remember (e.g., LetYesBig - capitalized for ease of reading -or how about ForEaseOf, or OfBigFree, or ...). Like I said, easy to remember, hard enough to guess.

Conclusions

Let's stop making these fear-mongering claims without proper stated assumptions, and start doing reasonable and prudent things instead.