Password Cracking Report

My computer running john the ripper on 4 OpenMP threads only gets around 4200 c/s on SHA-512 passwords. The expected number of attempts to crack an arbitrary string of length L with a character set of size N is given by the formula $(N^L)/2$. To crack an alphanumeric (N = 62) password of length 6 (L=6) it would take an average of 28,400,117,792 attempts (assuming a random distribution of characters). For this attack to be carried out by my computer it would take an estimated 28,400,117,792c / 4200c/s => 6,761,932 Seconds. This equates to ~2.5 months of cracking. For a password of 8 characters, the expected time is 824 years. For a password of 10 characters, it would take 3168334 years on average. Finally, for a password of 12 characters in length, it would take 12179077133 years, or 152 million human lifetimes.

I do not believe the password meter is a good indication of actual security. One password I tried to crack that was labeled as 'very weak' was 'Pass'. Cracking this was taking quite a while and I ended up killing the process after ~20 mins of attempts. Meanwhile, the password 'asdfasdf' was categorized as 'weak', a higher rating than 'Pass', yet 'asdfasdf' was cracked in less than a second. Clearly this online security estimator is merely an estimator, and cannot guarantee how difficult your password would be to crack. If you want your password to be secure against a state-level attacker, I would have a password which uses lowercase, uppercase, numbers and symbols and has length at least 14. For the password to be 'secure' I would also avoid using symbols as simple letter replacements, as that offers little to no security. For example, do not replace the letter a in 'awesome' with an @ sign, which would simply yield '@wesome'. I believe a nation state would be able to afford a million or so computers with high end equipment. If we assume each of these computers is using GPU power to accelerate their hashing, we could reason they could get around 33 billion hashes per second per computer. This would give a total hashing power of 3.3 * 10 ^ 16 per second. If you were to use a password of length 14 (L=14) with the proper character set, the alphabet would be of size 32 (symbols) + 62 (alphanumeric) (N=94), then it would take 65 billion seconds, or 2000 years to crack your password. For comparison, if you were to only use a password of length 12, this same nation state could crack your password in ~2.5 months. It is worth noting that humans are the weakest link in security. In all reality, a nation state could simply kidnap and torture you into giving up your password, making this whole analysis pointless.

For length 6, it would take an expected time of .86 seconds. For length 8, it would take an expected time of just under 2 hours. For length 10 it would take around 6 months. For length 12, it would take 1500 years. When answering #2, I already saw the numbers for hash rates, so I would leave my answer in its current state.

According to stack overflow, MD5 is slightly faster to compute than SHA-512, which makes offline guessing slightly faster. More importantly, SHA-512 is weakly (and strongly) collision resistant, which makes it much harder to find an input which will produce the desired hash. If MD5 is used, an attacker can trivially find a collision, then start trying that as a password. While finding a user's actual password is nearly as hard with MD5 as it is with SHA-512, finding a string which will function as a user's password is quite easy when MD5 is used. In summary, SHA-512 is more secure because it is slower and collision resistant.

Salts greatly improve password security. Firstly, by using a salt you cannot tell which users have matching passwords. This can reduce statistical attacks against passwords. Additionally, if other people have previously had their plaintext passwords compromised, you are not made less secure by using their same password. Secondly, salts are able to largely prevent rainbow table attacks where a hash is able to be looked up into plaintext through large amounts of precomputation. Rainbow table prevention is very important as it leads to exponentially more work for attackers trying to escalate their privileges.

Offline password attack protection is still immensely important. No matter what kind of security you think you have, it is always possible that you will suffer a breach. To minimize the exposure of your users, it is very important to prepare for that eventuality in whatever way you can. Through the use of industry recognized best practices and thorough validation, we can improve the privacy and security of those who use our services.