Anatomy of a hack: even your 'complicated' password is easy to crack

TECHNOLOGY 28 MAY 13
by DAN GOODIN

In March, readers followed along as Nate Anderson, Ars Technica deputy editor and a self-admitted newbie to password cracking, downloaded a list of more than 16,000 cryptographically hashed passcodes. Within a few hours, he deciphered almost half of them. The moral of the story: if a reporter with zero training in the ancient art of password cracking can achieve such results, imagine what more seasoned attackers can do.

Imagine no more. We asked three cracking experts to attack the same list Anderson targeted and recount the results in all their colour and technical detail Iron Chef style. The results, to say the least, were eye opening because they show how quickly even long passwords with letters, numbers, and symbols can be discovered.

The list contained 16,449 passwords converted into hashes using the MD5 cryptographic hash function. Security-conscious websites never store passwords in plaintext. Instead, they work only with these so-called one-way hashes, which are incapable of being mathematically converted back into the letters, numbers, and symbols originally chosen by the user. In the event of a security breach that exposes the password data, an attacker still must painstakingly guess the plaintext for each hash -- for instance, they must guess that "5f4dcc3b5aa765d61d8327deb882cf99" and "7c6a180b36896a0a8c02787eeafb0e4c" are the MD5 hashes for "password" and "password1" respectively. (For more details on password hashing, see the earlier Ars feature " Why passwords have never been weaker -- and crackers have never been stronger.")

While Anderson's 47-percent success rate is impressive, it's miniscule when compared to what real crackers can do, as Anderson himself made clear. To prove the point, we gave them the same list and watched over their shoulders as they tore it to shreds. To put it mildly, they didn't disappoint. Even the least successful cracker of our trio -- who used the least amount of hardware, devoted only one hour, used a tiny word list, and conducted an interview throughout the process -- was able to decipher 62 percent of the passwords. Our top cracker snagged 90 percent of them.

The Ars password team included a developer of cracking software, a security consultant, and an anonymous cracker. The most thorough of the three cracks was carried out by Jeremi Gosney, a password expert with Stricture Consulting Group. Using a commodity computer with a single AMD Radeon 7970 graphics card, it took him 20 hours to crack 14,734 of the hashes, a 90-percent success rate. Jens Steube, the lead developer behind oclHashcat-plus, achieved impressive results as well. (oclHashcat-plus is the freely available password-cracking software both Anderson and all crackers in this article used.) Steube unscrambled 13,486 hashes (82 percent) in a little more than one hour, using a slightly more powerful machine that contained two AMD Radeon 6990 graphics cards. A third cracker who goes by the moniker radix deciphered 62 percent of the hashes using a computer with a single 7970 card -- also in about one hour. And he probably would have cracked more had he not been peppered with questions throughout the exercise.

The list of "plains," as many crackers refer to deciphered hashes, contains the usual list of commonly used passcodes that are found in virtually every breach involving consumer websites. "123456," "1234567," and "password" are there, as is "letmein," "Destiny21," and "pizzapizza." Passwords of this ilk are hopelessly weak. Despite the additional tweaking, "p@$$word," "123456789j," "letmein1!," and "LETMEin3" are equally awful. But sprinkled among the overused and easily cracked passcodes in the leaked list are some that many readers might assume are relatively secure. ":LOL1313le" is in there, as are "Coneyisland9/," "momof3g8kids," "1368555av," "n3xtb1gth1ng," "qeadzcwrsfxv1331," "m27bufford," "J21.redskin," "Garrett1993*," and "Oscar+emmy2."

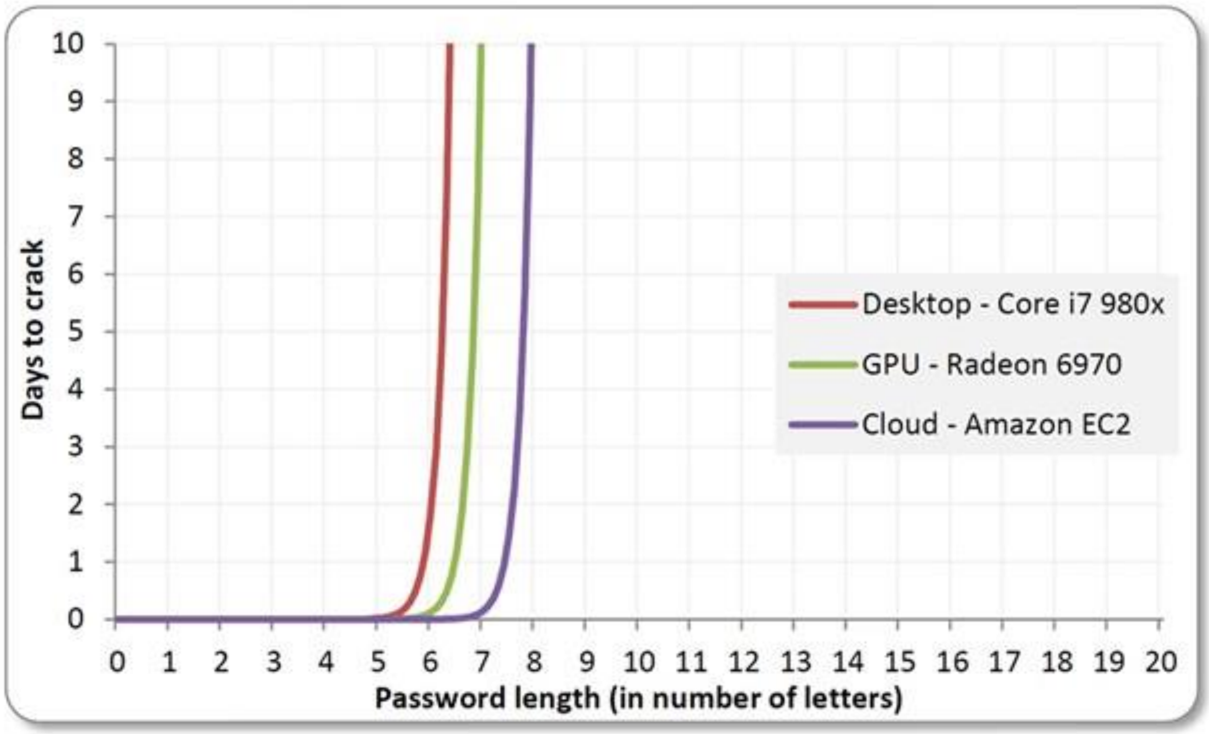
As big as the word lists that all three crackers in this article wielded -- close to 1 billion strong in the case of Gosney and Steube -- none of them contained "Coneyisland9/," "momof3g8kids," or the more than 10,000 other plains that were revealed with just a few hours of effort. So how did they do it? The short answer boils down to two variables: the website's unfortunate and irresponsible use of MD5 and the use of non-randomised passwords by the account holders.

Life in the fast lane
"These are terrible passwords," radix, who declined to give his real name, told Ars just a few minutes into run one of his hour-long cracking session. "There's probably not a complexity requirement for them. The hashing alone being MD5 tells me that they really don't care about their passwords too much, so it's probably some pre-generated site."

Like SHA1, SHA3, and most other algorithms, MD5 was designed to convert plaintext into hashes, also known as "message digests," quickly and with a minimal amount of computation. That works in the favour of crackers. Armed with a single graphics processor, they can cycle through more than eight billion password combinations each second when attacking "fast" hashes. By contrast, algorithms specifically designed to protect passwords require significantly more time and computation. For instance, the SHA512crypt function included by default in Mac OS X and most Unix-based operating systems passes text through 5,000 hashing iterations. This hurdle would limit the same one-GPU cracking system to slightly less than 2,000 guesses per second. Examples of other similarly "slow" hashing algorithms include bcrypt, scrypt, and PBKDF2.

The other variable was the account holders' decision to use memorable words. The characteristics that made "momof3g8kids" and "Oscar+emmy2" easy to remember are precisely the things that allowed them to be cracked. Their basic components -- "mom," "kids," "oscar," "emmy," and numbers -- are a core part of even basic password-cracking lists. The increasing power of hardware and specialised software makes it trivial for crackers to combine these ingredients in literally billions of slightly different permutations. Unless the user takes great care, passwords that are easy to remember are sitting ducks in the hands of crackers.

What's more, like the other two crackers profiled in this article, radix didn't know where the password list was taken from, eliminating one of the key techniques crackers use when deciphering leaked hashes. "If I knew the site, I would go there and find out what the requirements are," he said. The information would have allowed radix to craft custom rule sets targeted at the specific hashes he was trying to crack.



Brute-force cracks work well against shorter passwords. The technique can take days or months for longer passcodes, even when using Amazon's cloud-based EC2 serviceRob Graham, Errata Security

Anatomy of a crack
The longer answer to how these relatively stronger passwords were revealed requires comparing and contrasting the approaches of the three crackers. Because their equipment and the amount of time they devoted to the exercise differed, readers shouldn't assume one cracker's technique was superior to those of the others. That said, all three cracks resembled video games where each successive level is considerably harder than the last. The first stage of each attack typically cracked in excess of 50 percent of the hashes, with each stage that came later cracking smaller and smaller percentages. By the time they got to the latest rounds, they considered themselves lucky to get more than a few hundred plains.

True to that pattern, Gosney's first stage cracked 10,233 hashes, or 62 percent of the leaked list, in just 16 minutes. It started with a brute-force crack for all passwords containing one to six characters, meaning his computer tried every possible combination starting with "a" and ending with "//////." Because guesses have a maximum length of six and are comprised of 95 characters -- that's 26 lower-case letters, 26 upper-case letters, 10 digits, and 33 symbols -- there are a manageable number of total guesses. This is calculated by adding the sum of 95 to the power of 6 + 95 to power of 5 + […] + 95. It took him just two minutes and 32 seconds to complete the round, and it yielded the first 1,316 plains of the exercise.

Beyond a length of six, however, Gosney was highly selective about the types of brute-force attacks he tried. That's because of the exponentially increasing number of guesses each additional character creates. While it took only hours to brute-force all passwords from one to six characters, it would have taken Gosney days, weeks, or even years to brute-force longer passwords. Robert Graham, the CEO of Errata Security who has calculated the requirements, refers to this limitation as the "exponential wall of brute-force cracking."

Recognising these limits, Gosney next brute-force cracked all passwords of length seven or eight that contained only lower letters. That significantly reduced the time required and still cracked 1,618 hashes. He tried all passwords of length seven or eight that contained only upper letters to reveal another 708 plains. Because their "keyspace" was the sum of 26 to the power of 8 + 26 to the power of 7, each of these steps was completed in 41 seconds. Next, he brute-forced all passwords made up solely of numbers from one to 12 digits long. It cracked 312 passcodes and took him three minutes and 21 seconds.

It was only then that Gosney turned to his word lists, which he has spent years fine tuning. Augmenting the lists with the "best64" rule set built into Hashcat, he was able to crack 6,228 hashes in just nine minutes and four seconds. To complete stage one, he ran all the plains he had just captured in the previous rounds through a different rule set known as "d3ad0ne" (named after its creator who is a recognised password expert). It took one second to complete and revealed 51 more plains.

"Normally I start by brute-forcing all characters from length one to length six because even on a single GPU, this attack completes nearly instantly with fast hashes," Gosney explained in an email. He continued:

And because I can brute-force this really quickly, I have all of my wordlists filtered to only include words that are at least six chars long. This helps to save disk space and also speeds up wordlist-based attacks. Same thing with digits. I can just brute-force numerical passwords very quickly, so there are no digits in any of my wordlists. Then I go straight to my wordlists + best64.rule since those are the most probable patterns, and larger rule sets take much longer to run. Our goal is to find the most plains in the least amount of time, so we want to find as much low-hanging fruit as possible first.

Cracking the weakest passwords first is especially helpful when hashes contain cryptographic salt. Originally devised to thwart rainbow tables and other types of precomputed techniques, salting appends random characters to each password before it is hashed. Besides defeating rainbow tables, salting slows down brute-force and dictionary attacks because hashes must be cracked one at a time rather than all of them at once.

But the thing about salting is this: it slows down cracking only by a multiple of the number of unique salts in a given list. That means the benefit of salting diminishes with each cracked hash. By cracking the weakest passwords as quickly as possible first (an optimisation offered by Hashcat) crackers can greatly diminish the minimal amount of protection salting might provide against cracking. Of course, none of this applies in this exercise since the leaked MD5 wasn't salted.

With 10,233 hashes cracked in stage one, it was time for stage two, which consisted of a series of hybrid attacks. True to the video game analogy mentioned earlier, this second stage of attacks took considerably longer than the first one and recovered considerably fewer plains -- to be exact, five hours and 12 minutes produced 2,702 passwords.

As the name implies, a hybrid attack marries a dictionary attack with a brute-force attack, a combination that greatly expands the reach of a well-honed word list while keeping the keyspace to a manageable length. The first round of this stage appended all possible two-characters strings containing digits or symbols to the end of each word in his dictionary. It recovered 585 plains and took 11 minutes and 25 seconds to run. Round two appended all possible three-character strings containing digits or symbols. It cracked 527 hashes and required 58 minutes to complete. The third round, which appended all four-digit number strings, took 25 minutes and recovered 435 plains. Round four appended all possible strings containing three lower-case letters and digits and acquired 451 more passwords.

As fruitful as these attacks were, Gosney said they were handicapped by his use of a single graphics card for this exercise.

"For example, you'll notice that when I was doing hybrid attacks, I appended 2-3 digits/special but then only did digits with length 4," he explained. "This is because doing digits/special for length 4 would have taken a really long time with just one GPU, so I skipped it. Same with when I started appending lower alpha/digits, I only did length 3 because length 4 would have taken too long with just one GPU."

No doubt, Gosney could have attacked much larger keyspaces had he used the monster 25-GPU cluster he unveiled in December. Because the graphics cards in the five-server system scale almost linearly, it's able to harness almost all of their combined power. As a result, it can achieve 350 billion guesses per second when cracking password hashes generated by Microsoft's NTLM algorithm. And it could generate similar results when going up against MD5 and other fast hash functions.

The remaining hybrid attacks in stage two continued in the same vein. By the time it was completed, he had cracked a total of 12,935 hashes, or 78.6 percent of the list, and had spent a total of just 5 hours and 28 minutes doing it.

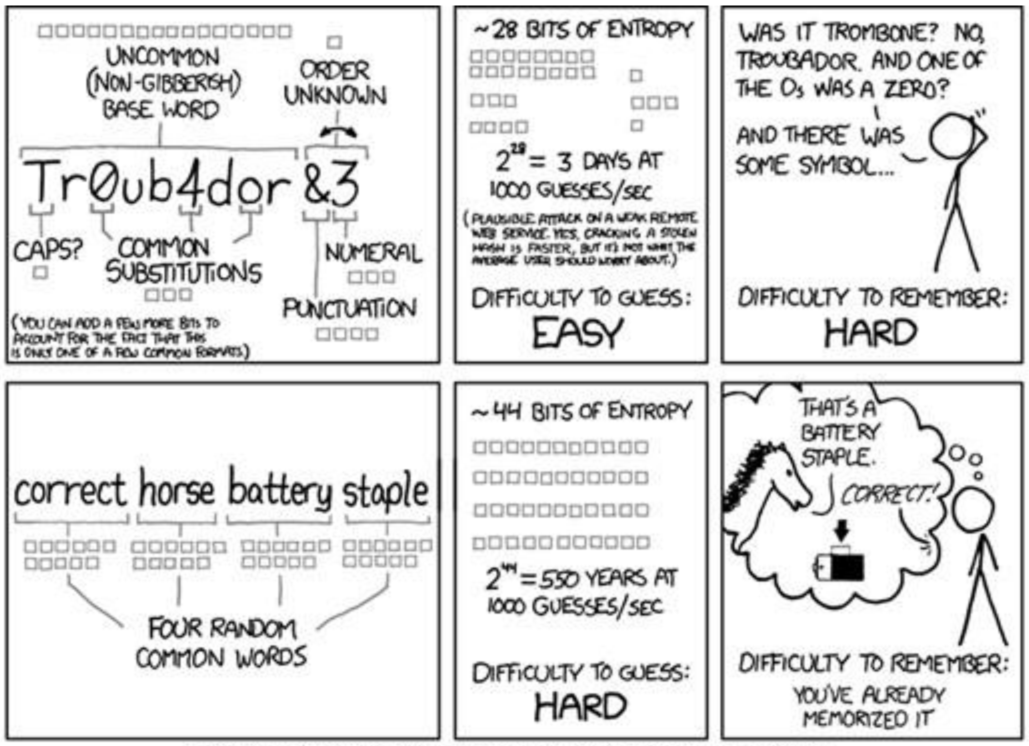
One of the things Gosney and other crackers have found is that passwords for a particular site are remarkably similar, despite being generated by users who have never met each other. After cracking such a large percentage of hashes from this unknown site, the next step was to analyse the plains and mimic the patterns when attempting to guess the remaining passwords. The result is a series of statistically generated brute-force attacks based on a mathematical system known as Markov chains. Hashcat makes it simple to implement this method. By looking at the list of passwords that already have been cracked, it performs probabilistically ordered, per-position brute-force attacks. Gosney thinks of it as an "intelligent brute-force" that uses statistics to drastically limit the keyspace.

Where a classic brute-force tries "aaa," "aab," "aac," and so on, a Markov attack makes highly educated guesses. It analyses plains to determine where certain types of characters are likely to appear in a password. A Markov attack with a length of seven and a threshold of 65 tries all possible seven-character passwords with the 65 most likely characters for each position. It drops the keyspace of a classic brute-force from 95 to the power of 7 to 65 to the power of 7, a benefit that saves an attacker about four hours. And since passwords show surprising uniformity when it comes to the types of characters used in each position -- in general, capital letters come at the beginning, lower-case letters come in the middle, and symbols and numbers come at the end -- Markov attacks are able crack almost as many passwords as a straight brute-force.

"This is where your attack plan deviates from the standard and becomes unique, because now you're doing site-specific attacks," Gosney said. "From there, if you start hitting upon any interesting patterns, you just start chasing those patterns down the rabbit hole. Once you've fully exploited one pattern you move on to the next."

In all, it took Gosney 14 hours and 59 minutes to complete this third stage, which besides Markov attacks included several other custom wordlists combined with rules. Providing further evidence of the law of diminishing returns that dictates password cracking, it yielded 1,699 more passwords. It's interesting to note that the increasing difficulty is experienced even within this last step itself. It took about three hours to cover the first 962 plains in this stage and 12 hours to get the remaining 737.

The other two password experts who cracked this list used many of the same techniques and methods, although not in the same sequence and with vastly different tools. The only wordlist used by radix, for example, came directly from the 2009 breach of online games service RockYou. Because the SQL-injection hack exposed more than 14 million unique passwords in plaintext, the list represents the largest corpus of real-world passwords ever to be made public. radix has a much bigger custom-compiled dictionary, but like a magician who doesn't want to reveal the secret behind a trick, he kept it under wraps during this exercise.


UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor&3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)
~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS: EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER: HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT

THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

Killing hashes
Like Nate Anderson's foray into password cracking, radix was able to crack 4,900 of the passwords, nearly 30 percent of the haul, solely by using the RockYou list. He then took the same list, cut the last four characters off each of the words, and appended every possible four-digit number to the end. Hashcat told him it would take two hours to complete, which was longer than he wanted to spend. Even after terminating the run two after 20 minutes, he had cracked 2,136 more passcodes. radix then tried brute-forcing all numbers, starting with a single digit, then two digits, then three digits, and so on (259 additional plains recovered).

He seemed to choose techniques for his additional runs almost at random. But in reality, it was a combination of experience, intuition, and possibly a little luck.

"It's all about analysis, gut feelings, and maybe a little magic," he said. "Identify a pattern, run a mask, put recovered passes in a new dict, run again with rules, identify a new pattern, etc. If you know the source of the hashes, you scrape the company website to make a list of words that pertain to that specific field of business and then manipulate it until you are happy with your results."

He then ran the 7,295 plains he recovered so far through PACK, short for the Password Analysis and Cracking Toolkit (developed by password expert Peter Kacherginsky), and noticed some distinct patterns. A third of them contained eight characters, 19 percent contained nine characters, and 16 percent contained six characters. PACK also reported that 69 percent of the plains were "stringdigit" meaning a string of letters or symbols that ended with numbers. He also noticed that 62 percent of the recovered passwords were classified as "loweralphanum," meaning they consisted solely of lower-case letters and numbers.

This information gave him fodder for his next series of attacks. In run 4, he ran a mask attack. This is similar to the hybrid attack mentioned earlier, and it brings much of the benefit of a brute-force attack while drastically reducing the time it takes to run it. The first one tried all possible combinations of lower-case letters and numbers, from one to six characters long (341 more plains recovered). The next step would have been to try all combinations of lower-case letters and numbers with a length of eight. But that would have required more time than radix was willing to spend. He then considered trying all passwords with a length of eight that contained only lower-case letters. Because the attack excludes upper case letters, the search space was manageable, 26 to the power of 8 instead of 52 to the power of 8. With radix's machine, that was the difference between spending one hour and six hours respectively. The lower threshold was still more time than he wanted to spend, so he skipped that step too.

So radix then shifted his strategy and used some of the rule sets built into Hashcat. One of them allows Hashcat to try a random combination of 5,120 rules, which can be anything from swapping each "e" with a "3," pulling the first character off each word, or adding a digit between each character. In just 38 seconds the technique recovered 1,940 more passwords.

"That's the thrill of it," he said. "It's kind of like hunting, but you're not killing animals. You're killing hashes. It's like the ultimate hide and seek." Then acknowledging the dark side of password cracking, he added: "If you're on the slightly less moral side of it, it has huge implications."

Steube also cracked the list of leaked hashes with aplomb. While the total number of words in his custom dictionaries is much larger, he prefers to work with a "dict" of just 111 million words and pull out the additional ammunition only when a specific job calls for it. The words are ordered from most to least commonly used. That way, a particular run will crack the majority of the hashes early on and then slowly taper off. "I wanted it to behave like that so I can stop when things get slower," he explained.

Early in the process, Steube couldn't help remarking when he noticed one of the plains he had recovered was "momof3g8kids."

"This was some logic that the user had," Steube observed. "But we didn't know about the logic. By doing hybrid attacks, I'm getting new ideas about how people build new [password] patterns. This is why I'm always watching outputs."

The specific type of hybrid attack that cracked that password is known as a combinator attack. It combines each word in a dictionary with every other word in the dictionary. Because these attacks are capable of generating a huge number of guesses -- the square of the number of words in the dict -- crackers often work with smaller word lists or simply terminate a run in progress once things start slowing down. Other times, they combine words from one big dictionary with words from a smaller one. Steube was able to crack "momof3g8kids" because he had "momof3g" in his 111 million dict and "8kids" in a smaller dict.

"The combinator attack got it! It's cool," he said. Then referring to the oft-cited xkcd comic, he added: "This is an answer to the batteryhorsestaple thing."

What was remarkable about all three cracking sessions were the types of plains that got revealed. They included passcodes such as "k1araj0hns0n," "Sh1a-labe0uf," "Apr!l221973," "Qbesancon321," "DG091101%," "@Yourmom69," "ilovetofunot," "windermere2313," "tmdmmj17," and "BandGeek2014." Also included in the list: "all of the lights" (yes, spaces are allowed on many sites), "i hate hackers," "allineedislove," "ilovemySister31," "iloveyousomuch," "Philippians4:13," "Philippians4:6-7," and "qeadzcwrsfxv1331." "gonefishing1125" was another password Steube saw appear on his computer screen. Seconds after it was cracked, he noted, "You won't ever find it using brute force."

The ease these three crackers had converting hashes into their underlying plaintext contrasts sharply with the assurances many websites issue when their password databases are breached. In April, when daily coupons site LivingSocial disclosed a hack that exposed names, addresses, and password hashes for 50 million users, company executives downplayed the risk.

"Although your LivingSocial password would be difficult to decode, we want to take every precaution to ensure that your account is secure, so we are expiring your old password and requesting that you create a new one," CEO Tim O'Shaughnessy told customers.

In fact, there's almost nothing preventing crackers from deciphering the hashes. LivingSocial used the SHA1 algorithm, which as mentioned earlier is woefully inadequate for password hashing. He also mentioned that the hashes had been "salted," meaning a unique set of bits had been added to each users' plaintext password before it was hashed. It turns out that this measure did little to mitigate the potential threat. That's because salt is largely a protection against rainbow tables and other types of precomputed attacks, which almost no one ever uses in real-world cracks. The file sizes involved in rainbow attacks are so unwieldy that they fell out of vogue once GPU-based cracking became viable. (LivingSocial later said it's in the process of transitioning to the much more secure bcrypt function.)

Officials with Reputation.com, a service that helps people and companies manage negative search results, borrowed liberally from the same script when disclosing their own password breach a few days later. "Although it was highly unlikely that these passwords could ever be decrypted, we immediately changed the password of every user to prevent any possible unauthorised account access," a company email told customers.

Both companies should have said that, with the hashes exposed, users should presume their passwords are already known to the attackers. After all, cracks against consumer websites typically recover 60 percent to 90 percent of passcodes. Company officials also should have warned customers who used the same password on other sites to change them immediately.

To be fair, since both sites salted their hashes, the cracking process would have taken longer to complete against large numbers of hashes. But salting does nothing to slow down the cracking of a single hash and does little to slow down attacks on small numbers of hashes. This means that certain targeted individuals who used the hacked sites -- for example, bank executives, celebrities, or other people of particular interest to the attackers -- weren't protected at all by salting.

The prowess of these three crackers also underscores the need for end users to come up with better password hygiene. Many Fortune 500 companies tightly control the types of passwords employees are allowed to use to access email and company networks, and they go a long way to dampen crackers' success.

"On the corporate side, its so different," radix said. "When I'm doing a password audit for a firm to make sure password policies are properly enforced, it's madness. You could go three days finding absolutely nothing."

Websites could go a long way to protect their customers if they enforced similar policies. In the coming days, Ars will publish a detailed primer on passwords managers. It will show how to use them to generate long, random passcodes that are unique to each site. Because these types of passwords can only be cracked by brute force, they are the hardest to recover. In the meantime, readers should take pains to make sure their passwords are a minimum of 11 characters, contain upper- and lower-case letters, numbers, and letters, and aren't part of a pattern.

The ease these crackers had in recovering as many as 90 percent of the hashes they targeted from a real-world breach also exposes the inability many services experience when trying to measure the relative strength or weakness of various passwords. A recently launched site from chipmaker Intel asks users "How strong is your password?," and it estimated it would take six years to crack the passcode "BandGeek2014". That estimate is laughable given that it was one of the first ones to fall at the hands of all three real-world crackers.

As Ars explained recently, the problem with password strength meters found on many websites is they use the total number of combinations required in a brute-force crack to gauge a password's strength. What the meters fail to account for is that the patterns people employ to make their passwords memorable frequently lead to passcodes that are highly susceptible to much more efficient types of attacks.

"You can see here that we have cracked 82 percent [of the passwords] in one hour," Steube said. "That means we have 13,000 humans who did not choose a good password." When academics and some websites gauge susceptibility to cracking, "they always assume the best possible passwords, when it's exactly the opposite. They choose the worst."

http://arstechnica.com/security/2013/05/how-crackers-make-minced-meat-out-of-your-passwords/