

The password arms race

Eric Blair

25 March 2008

Economists tend to think of people as a lazy, pernicious ooze. If there's a crack in a system, somebody will ooze their way into it and exploit the flaw. After the easy and profitable flaws are covered, people may need to move on to more difficult ways of being lazy.

Password cracking fits this mold: a cracker doesn't crack all the passwords, but just finds the easiest chink in the system and attacks that. If there's one person in the company whose password is *password*, they're in.

Recently, two data sets of passwords from real people came out; see Schneier and this writeup. It turns out that, if given the chance, an awful lot of people really will use *password* as their password.

Sysadmins are playing a complementary game to the crackers. They guess the most common passwords, and then deny you access until you can come up with a password that gets past their barriers. And of course, the users, being a pernicious and lazy ooze, invent the absolute simplest password that matches the requirements of the system. OK, *password* is out, so use *password1*.

In the beginning, you'd be prompted to just enter a password. The lowest-effort person just used *password*. Thus, we had the first round of the escalation, which went something like this:¹

Your password will be checked against a 'hacker dictionary' of English words, common names, and common product brands, and will be rejected if we find it in the dictionary. Please make an effort to mix caps, numbers, and symbols. If you use *password* as your password, HR will be notified and you will be asked to clear out your desk.

Things continued to escalate from there, and every year we saw more clauses and conditions added. Here are the seven rules from my telephone company login screen. I left them in Spanish to keep them extra inscrutable [If you can't read this, you're experiencing what a huge chunk of the Internet's account setup screens look like to most of the world's users.]:

Tu contraseña debe reconocer las mayúsculas y:

- Debe tener entre seis y veinte caracteres.
- No emplear otros caracteres que no sean letras y números (tales como *, &, # y """).

¹I never really saw a prompt like this, but you'll see below that it's sort of my ideal balance.

- No utilizar la fecha de nacimiento, el nombre o el apellido ni la combinación del nombre y el apellido.
- No incluir los primeros cuatro a ocho dígitos de tu número de teléfono móvil.
- No debe coincidir con tu número de cuenta.
- No debe ser una dirección de correo electrónico.
- No debe tener más de 2 caracteres repetidos a continuación ni más de 3 caracteres ascendentes consecutivos.

You can see that this escalation can go on forever: users use *password*; so admins banned it; users lazily ooze to *password1*; so admins ban it by requiring numbers, lowercase, and capitals; users ooze to *Password1*, et cetera. From the list of rules, you can almost guess the most common password, which means that the rules don't really bring up the difficulty crackers will have in guessing that bottom end.

Oh, and some systems are broken, like the second bullet point above that won't let you use any characters but letters and numbers. Seriously? Your system can't encrypt and store an arbitrary string of text? Some require a password that is exactly six, seven, or eight characters long, which is just inexcusably lazy database design. So those details add another level of password complexity, as it's called.

Password complexity This is really a column about how organizations wind up with technical rules like the above, but let's pause to look at the underlying math. As far as I can tell, the term *password complexity* refers to the size of the space in which your password lives. *Common personal/account characteristics* is probably only maybe two dozen elements (your first name, your last name, your dog's name, ...). *English words* is tens of thousands of words: OpenOffice.org's spell-checking dictionary² has several English variant dictionaries, which each are in the ballpark of 50,000 words. [I tried to use Aspell's dictionary, but couldn't find the dictionary source anywhere.] But *arbitrary combinations of English letters* is an absolutely huge space: for five letters, $26 \times 26 \times 26 \times 26 \times 26 = 11,881,376$. By eight letters, you've hit two hundred billion. Of course, our password requirements like to take it further. With *26 letters plus 10 numbers*, you have 60 million five-character passwords and 2.8 trillion eight-character combos.

Indeed, 2.8 trillion is about fourteen times bigger than 200 billion. But by 200 billion, the game is already over: who cares if your space is four million times bigger than the English dictionary or sixty million times bigger? You've already locked out any casual attempts and script kiddies, and the truly sophisticated will get around the larger requirements too. [True story: a pal offered to share wireless with her neighbor David. He never mentioned anything again, but one day a password protected router named apt302 turned up on her computer's list. The first password she tried was *Dave*, and she was in.]

Once you've left the English dictionary, you've done as much as reasonable for dictionary-based attackers. As for the lazy and pernicious ooze problem, a larger space doesn't help at all: there will always be an effort-minimizing corner somewhere. So for this detail of the issue, the added restrictions are just a waste, and the only serious solution is to educate users that they have to stop meeting all the requirements with crap like *PasswOrd!*.

²<http://wiki.services.openoffice.org/wiki/Dictionaryes>

Me, I like the First Letter Of A Phrase method—the floap method. Almost any sentence will produce a non-English string of letters which is easy to remember and has much higher ‘complexity’ than any system that arrives at a dictionary word with a 1 appended and all the ‘o’s changed to zeros. It’s certainly hard to guess: how many phrases and catchy pop songs do you know? I’ve written up this rant, though, because even the floap method is increasingly not acceptable: e.g., many systems will tell me that *floap* is a bad password, having only five letters, no capitals, and no numbers. But it’s already outside of the English dictionary, and is thus in the space of 11 million possibilities and not fifty thousand.

We need to pull people out of the 50,000-word dictionary and into using something more complex, but burdening them with extra rules to move them to a space of 2 billion options instead of 200 million is likely counterproductive. At work, somebody may have to spend ten minutes playing Scrabble with the password-checker instead of doing productive work, only to forget the password and get locked out the next day. Online, every additional password requirement is another bunch of people who just closed the browser window and went somewhere else.

Social I searched Web of Science for academic journal articles about password complexity like the Byzantine list of requirements above. I found none. [Feel free to mention any that I missed in the comments.] The concept of password complexity is, as far as I can tell, pure managementspeak.

After all, the combinatorics above already cover most of the story: the English dictionary is short, and random character streams are hard to search. So why continue the escalation, then, adding longer and more absurd requirements every year?

First, the person setting the password rules looks like s/he is doing due diligence by having password rules as annoying or more so than the neighbors’. If the rules are weaker and there’s a crack, then it’s the admin’s fault; if the rules are absurdly onerous and there’s still a crack, then the admin’s ass is covered. So there’s a simple Prisoner’s Dilemma game, where we all collectively benefit from keeping norms sane, yet admins want to signal that their rules are more impressive than the norm.

That signal is pretty public, too. When you first log in and set up your password, that list of silly rules is a big sign from the IT department stating that they are deeply concerned with security and must be doing lots of things under the hood that are just as carefully considered as this long list of rules. It’s only natural that admins would go crazy and overthink the user-facing side of things, whether the effort benefits the system or not.

The potential organizational costs of a breach of security are cleanly focused on the admin who would get fired. The cost of ten minutes wasted by every person in the organization trying to come up with a password, forgetting it, getting locked out of their work, and so on, is by definition a distributed cost. So there’s a collective action problem to the situation: how can we get a balanced consideration of the admin’s concentrated interest against the inconvenience of every user? At one place I work, being locked out is so prevalent that there’s a specially-built site for password resetting, iforgotmypassword....gov. I take this to mean that the balance has not been struck, and the password complexity rules are too burdensome [I wanted to post them for you, but the site

only works in Windows, and even attempts to get a list of the rules give me errors from IIS.]

And, of course, there's the mystic nature of security. *It's a security issue* wins every argument for your IT guy, and s/he knows it. How do you argue with that? They know the details and you don't, and even if you did know the details, you'd realize that security can't actually be measured anyway.