Non-Electronic Password Generator/Recall

Design Considerations and A Threat Model Discussion

January, 2016 [Update: September, 2020]
Russell Magee
Russtopia Labs

Introduction and Rationale

This document attempts to address questions of a more technical nature for those who are skeptical or merely curious about the design of the c@rd™ (and related) password generator/recall products. Amongst other things, it details the impetus for the design of the

product(s) and the niche that is filled: that which lies between the extremes of no formal password management regime at all, and modern two-factor or multi-factor password management systems which require complex hardware user tokens and/or third-party infrastructure in order to function.

Also discussed is what the product *is not* designed for -- what uses are inappropriate -- in order to help the user understand how to gain some modicum of benefit from it whilst preventing a false sense of security.

Finally, a more technical discussion of the security primitives used in the design of the c@rd™ (and related products) is presented in order to illustrate how its design and production seek to thwart various types of attacks.

Framing the Discussion

These products do not solve the world's password problems, for once and ever, and deliver a victory bag of chips and a beer afterwards. Like any tool, they have their uses and *mis*-uses. Understanding what the products are, and are not, is essential to using them properly.

Target Audience

The target audience of this product is *not* someone who feels at ease using online password-management systems such as LastPass, encrypts email using GPG, and reads CERT advisories for fun (who doesn't, right?). Those people (should) know what makes for strong passwords, the dangers of password reuse, and how to practise good password habits in order to minimise the chances passwords will be lost, forgotten, or compromised.

The target audience *is* someone without a desire, knowledge, or ability to use the above products and procedures, who might think a CERT advisory has something to do with breath mints (sorry, couldn't resist), who nonetheless suspects their password habits are somehow *bad* -- and wants to do *something* meaningful to improve them. They want stronger passwords than what they presently use, but aren't sure how to achieve that. They are afraid they'll forget their passwords if they try to make them more complex. They need something to lighten their cognitive load when creating, and later recalling, complex passwords.

Changing the World Is Hard

For most, remembering more than a few passwords, and which ones are used where, is difficult. As a result most people just give up on the problem, and -- on average -- use 5 or fewer passwords across 26 or more accounts.

To highlight and (*unsuccessfully*) combat the problem, the security industry has tried:

- Publicly shaming the world by doing things like publishing an annual "Top 10,000 Worst Password" list (which 99.9% of the world never reads and blissfully ignores);

- Insisting that everyone needs to use electronic password wallet systems (which 99.9% of the world has *also* blissfully ignored, or actively rejected; and besides these tools have their own share of vulnerabilities -- see here and here);

- Proclaiming that single-factor password authentication itself must be abolished worldwide. Good luck with coordinating *that* roll-out -- consider that COBOL still exists. Anyone who has ambitions to convince every online provider to update password authentication schemes in *anybody's* lifetime… not likely to happen;

- Proclaiming that anything less than 2FA (two-factor auth) or MFA (multi-factor auth) might as well be nothing at all, so don't bother! This is a False Dilemma argument and a defeatist *non-solution* for regular people who don't have access to such systems and *must* use websites which don't support 2FA/MFA. Besides, MFA isn't perfect either;

- Unilaterally disabling single-factor authentication on the (few) systems they control, annoying their users to the point of avoiding or outright abandoning their services. XBox Live users will know the pain of trying to go through its multi-factor authentication to update a password. It ain't pretty. I guess that's one way to solve password security for a system: make it so onerous to use that people will just *stop using your system*. Believing that *your* system is indispensable is pure hubris (consider that Myspace was once the largest social media platform).

The reality is, many non-technical users -- not being used here as a pejorative term, but rather as an objective description -- do not relish the idea of using *yet another* fragile, expensive, and hard-to-replace electronic device and do not want to fire up *yet another* program or web service in order to use some electronic password wallet, like LastPass.

Go ahead -- ask your relatives about LastPass. Bask in the resulting blank stare.

The other reality to be considered is that these high-tech attempts at better password security have their own unique issues and, by the very nature of their centrality, make for uniquely high-value targets.

Online, PC- or mobile-based password management systems open themselves to an entire class of vulnerabilities *just by being connected to a network* or running on a computer connected to a network. As of early 2016, a critical attack against the popular LastPass service was identified. Even the vaunted RSA SecurID system has been compromised on more than one occasion, requiring massive recalls of expensive ID tags. Nothing's perfect.

Denying Human Nature

The average user does not want to be inconvenienced much. That is unfortunate, but one can't deny human nature. This is one of the key reasons why everyone doesn't already use multi-factor authentication; why companies offer one-click 'buy' buttons; and why companies seldom use self-signed certificates on their sites (self-signed certs cause warning popups which scare off potential customers, so few sites use them, security be damned!) and why passwords still are required.

One can argue until one is blue in the face why this should not be, but it isn't going to change any time soon.

A Compromise

If people refuse to be inconvenienced beyond a certain point, then it makes sense to offer something that will give at least *some* additional security whilst remaining convenient enough that it might, you know, *actually be used*.

Security comes in degrees. These devices offer a middle-ground of security -- the use of which can be easily taught to non-technical people and, while not as secure in a formal sense as multi-factor authentication or other electronically-assisted systems, is at least a step up from using no assistance at all (*'superman123' everywhere*), or trivial variations consisting of only a few characters (*'sup3rm4n123', 'superman124', or 'gmailsux', 'y0utub3sux', …*).

They can also offer an additional layer of defence for those more technical users who wish to enhance their already in-place password management systems (for example, having multiple password wallets, and wishing to use separate master passwords for each).

What the products ARE NOT

They are *not* a device that magically makes your passwords unguessable, without any mental effort at all. They *do not* guarantee, 100%, that *every* password you make using them will be unique, unless you've come up with a rule that is essentially a mentally-performed *perfect hash function*; in which case, congratulations, you are likely a mathematical prodigy and/or possess an eidetic memory and have no need for any sort of password manager system, including the ones discussed here.

They are *not* as secure against impersonation/cloning attacks as two-factor challenge/response authentication systems such as RSA SecurID, and the passwords one generates with them may not be *quite* as long or quite as random as those generated by password managers like LastPass.

What the products ARE

They *are*, however, a memory aid, and a system to be used to help one make (and recall) *better* passwords than what could likely be generated and memorised unaided. They *are* a way to make *most* of one's passwords different between the dozens of accounts one may have.

They *are*, unlike multi factor authentication systems, able to be used across multiple systems without unified login infrastructure and, unlike systems like LastPass, are not vulnerable in the same manner to browser impersonation attacks, or attacks which try to capture or enumerate multiple master passwords, as there is no central storage or master password for one to lose. There is no need to connect to a remote system, or to have access to a particular PC or mobile device, in order to store or recall passwords.

Using Securely

In order to gain the intended benefit from the product, some simple principles must be kept in mind by the user.

The Short Version

- Keep the device in possession and concealed unless in active use;
- Keep the dial-in or lookup rule secret;
- Use a rule that minimises the likelihood of different accounts ending up with the same password.
- Use a rule that is *not* the example rule shipped with the product, preferably not quite as simple as the example rule. Be creative.

The Long Version

If you let someone photograph your device or otherwise take a complete copy, then yes, you've reduced the security of your passwords (depending on the ratio of the length of your secret word/phrase to the length of symbols read off of your device, this could be a minor or a major reduction).

If that same someone goes on to attack *your* passwords specifically, it would be as if you weren't using the device at all and just using your secret word/phrase on all your accounts.

If you tell someone your rule but keep your device confidential, you've also reduced the security of your passwords in the respect that they now know the 'memorable' part of your passwords, and only have to brute-force guess the randomised parts of each.

If you tell someone your rule *and* let them copy/photograph your device, you've eliminated your password security completely.

Buying a helmet won't save you in a bike accident if you leave it in the closet at home -- you must use the tool consistently and correctly in order to gain any benefit from it. So note the above, and always wear your helmet, so to speak :p

Keep your card at *least* as concealed as your driver's licence or your health card -- in your pocket except when you need to use it. It has PII (Personally-Identifying Information). If you lose it, it should be replaced with a different one, just like a credit card.

Never tell anyone your *mental rule*. If someone asks how the device works, show them the *example rule* in the product documentation, *not* your own.

Make your rule something that isn't *too* likely to give you the same password fragment for multiple accounts. For example, if you did use the example rule in the documentation, using the first two letters of a website's domain as columns to look up symbols, then you'll get the same password fragment for 'ford.com', 'forbes.com', and 'foldablebananas.org'.

This may be an acceptable risk to you. You must use your judgement when creating a rule to use with the device, knowing your rule may introduce a risk of name collisions.

Again, this is a *tool*. It's up to you to use it intelligently to maximise its effectiveness in giving you as many different passwords as is practical. Even if you do end up with, say, 2 or 3 accounts out of 30 that have the same password, that's *a lot better* than most people's passwords (who, again, on average, have less than 5 different passwords for *everything*).

A Dose of Realism: How Much Password Security Do You Need?

Perhaps you're an honest-to-gosh actual Security Professional™ whose responsibilities make you and your systems a target for über-hackers. Perhaps you have concerns that your activities merit special 'TLA' attention. This product is *not for you*. You should make arrangements with a company that offers multi-factor authentication infrastructure, become your own certificate authority (or use self-signed certs if you don't trust the cert authority system itself) and do everything over VPNs and Tor. You should also consider meeting in dark alleys with those one or two really sensitive contacts of yours and physically exchange a stack of one-time-pads.

If your 'threat model' -- the types of attacks you're really worried about in day-to-day life -- *really do* include secret agents following you in unmarked vans with cameras and going through your garbage, then this product is *not for you*. You have **much bigger** problems and multi-factor authentication with dongles, fingerprint and iris scanners won't protect your accounts from someone *that* determined. These products weren't designed for that.

What they *were* designed for is to make life difficult for some random hacker, far away, who doesn't know you, John Q. Public. They don't know you from the other 7 billion-odd people on the planet -- save for, possibly, the website(s) you visit, your name, your favorite superhero, what company you work for; and, more importantly, the fact that your password (or a badly-stored hash, or even plaintext version of it) is stored in that juicy database they've just

obtained by hacking some company website. They're not trying to hack ***you*** **specifically**… they're just looking for weak and re-used passwords to gain access to email or social network accounts so they can spam the world, or maybe send your contacts some ransomware links.

If they are determined enough, they *will* crack the majority of that database, containing a single password of yours. But if you've used your device according to instructions, that *particular* password might not be cracked by the heuristics they're using. Even if they *do* crack it, it *won't* be the same one you use everywhere else, so it won't expose any of your other accounts. *And they weren't targeting you specifically*, so they won't be applying any sort of deeper analysis on that *one* compromised password in order to crack the others you generated using this device for the other sites they don't know about.

Far too many people just use passwords like 'superman123'. Or 'hotmailsucks' (for their hotmail account! Sheesh). And they use it across most, or **all** of their accounts. Those types of passwords are what hackers love to find, and what the c@rd™ (and other related products) were intended to help prevent.

The c@rd™ Security Model

As stated above, the security of *one* c@rd, and the passwords it helps the user create, is highly dependent on the user's habits.

But what of the manufacturing system as a whole: is the system of card issuance itself subject to attack? That is, given one or more cards, can a malicious party impersonate the manufacturer and generate arbitrary cards independently, which would be identical to the ones legitimately issued?

This turns out not to be the case, as the security and uniqueness of each card lies in the hash algorithm and salting used to seed the PRNG.

c@rd™ Generation

The contents of each card are based on three quantities, concatenated as input into a hashing algorithm, to derive a digest which is then used as the seed to the PRNG:

- The 'client ID': printed on the card
- An 'issue code', usually, but not exclusively a number: also printed on the card
- An 'issuing authority ID', a very long random string, much longer than both the client ID and the issue code, and NOT printed on the card.

The hashing algorithm chosen is SHA2-256, a well-regarded hashing algorithm with very strong anti-collision properties and specifically designed to resist attacks which attempt to craft inputs

intended to produce a specific output digest value. It was designed to address weaknesses which became apparent in SHA1 after its introduction, with the goal to resist attacks for the foreseeable future, even accounting for the typical rate of increase in computing power over time.

The PRNG used is Mersenne Twister, which is also very well-regarded. It is outside the scope of this document to describe Mersenne Twister, but suffice it to say it is the PRNG used by most modern programming languages due to its qualities.

Issuing Authority Security Considerations

The system used by the issuing authority ID to generate cards should be run offline (air-gapped) or at least on a suitably firewalled and isolated network, requiring a trusted individual to enter the issuing authority ID interactively to authenticate each batch of production. Ideally the issuing authority ID is not stored online, instead being entered from offline media by the trusted individual producing the batch.

As with any salt input to a hash, it should be difficult to guess by an attacker, and protected at all costs by the issuing authority, akin to an organisation's certificate signing key.

c@rd™ Cloning Attack Resistance

The difficulty of reversing a SHA2-256 hash, given incomplete knowledge of the input, is the key to preventing unauthorised, malicious reproductions of someone else's card. Independent generation of a card by guessing the issuing authority ID, starting only from the visual contents of a legitimately issued card, would imply computing a SHA2-256 collision, which if achieved in less than a brute-force number of computations would demonstrate a break of SHA2-256 itself.

Conclusion

While multi-factor authentication, using challenge/response channels with trusted tokens (electronic or biometric) beyond the primary password authentication mechanism, in combination with third-party or in-house infrastructure is the current state-of-the-art in security, it has in practice not been effective within and without corporate and organizational settings due to the lack of universal standardization and the technical complexity and training logistics involved in using and deploying such schemes. An offline scheme which offers an incremental improvement over naive password policies used by the general public is, while not ideal, better than an unrealistic ideal for security.

A reasonable parallel is illustrated in this paper regarding the ad-hoc deployment of SSH as a replacement for telnet, rcp and other utilities in previous decades -- which was wildly successful

despite its imperfections:

- **A professional cryptographer would have designed a system around certificates issued by properly-isolated and secured CAs**
- **In a very real sense, that would have been more secure — and it would likely have been undeployable**
- **We got more real security from a partially-secure implementation that better matched deployment patterns**