Password Synthesizers vs. Password Managers

Francisco Ruiz, MMAE Department
Illinois Institute of Technology
ForenSecure 2019
Fake news?

DoD to require passwords to be changed every day, use at least 27 different letters

Published February 6, 2017
By Phil McCracken

FORT MEADE, Md. – The Pentagon has announced new password requirements to beef up security on all DoD networks, Duffel Blog has learned. According to a recent memo, the DoD has made it a top priority to protect members’ Personally Identifiable Information (PII) by using the following requirements:

• Passwords must be at least 54 characters but not more than 56 characters long
• 27 characters must be non-repeating letters
• 19 characters must be numbers that cannot repeat, be odd, or be even
• The time allowed to enter the password is between 3 and 5.7 seconds
• The password must be entered at high tide
• Users will be required to reset their password after one correct attempt
And of course,

• Different passwords for different logins
• Long (at least 8 characters, per NIST SP 800-63B-3 2018 guidelines)
• Including different kinds of characters
• But not including dictionary words (NIST SP 800-63B-3)
• Changed with some frequency (forced change deprecated by NIST SP 800-63B-3)
• Hard to guess
• Easy to remember?
How to remember passwords

• Same everywhere
• Never changed
  • Not recommended, and increasingly hard to get away with
• Stored
  • Paper
  • Digital: Password Manager
• Generated on the fly
  • Password Synthesizer
Managers vs. Synthesizers

Manager
• Works by storing
• User comes up with password
• Usually not random-looking
• Backend at local app or internet

Synthesizer
• No storage
• App generates password
• Always random-looking
• No backend
Some Examples

Managers
• LastPass
• 1Password
• KeePass
• Chrome (built-in)
• Apple keychain
• ....

Synthesizers
• SynthPass
• Master Password
• PasswordGen
• PwdFly
• PawHash
• ....
How a manager works

To store password
1. Loaded extension looks for filled password fields; triggers popup upon submission
2. If user agrees, password is encrypted with Master Password (may trigger new popup) and added to database, listed by page origin
3. Database is synced locally or over internet

To fill password
1. User clicks button, enters Master Password
2. Extension looks for password fields in page, looks for existing entry in database (supplied by local app or over internet)
3. If found, pops up to fill after local decryption
How a synthesizer works

To store password:
• Nothing. There is no storage

To fill password:
1. User clicks button, enters Master Password
2. Extension combines Master with page origin, performs key derivation function (multiple-round hash)
3. Output is displayed to be copied (typical), or fills password field in page (SynthPass)
Manager pros and cons

**Pros**
- Remembers stuff
- Detects password changes
- Wrong Master detected
- Mobile apps
- Master compromise not fatal

**Cons**
- Requires secure backend
- Doesn’t enforce strong password
- Intrusive popups
- Duplicates
- Hard to sync devices
- Subscription cost
Synthesizer pros and cons

**Pros**
- No backend
- Always high strength
- Always in sync
- Free

**Cons**
- Does not remember stuff
- Hard to change password
- Varying password policies?
- What if Master is wrong?
- No mobile
- Master compromise is fatal
How bad can it get?

1Password user for a few years

• About 1/3 of entries have duplicates, resulting from:
  • Change of password
  • Login at different machines
  • Gremlins?

• When client changed, I had 2 different vaults accessed by my devices:
  • Syncing nightmare

• And this is one of the best
  • Vault is local, LastPass in the Cloud
So I made my own. . .

It’s a synthesizer, but:

1. Does remember stuff
   • User ID, password length and special characters
2. Designed for password changes
   • Optional serial, which is also remembered
   • Multiple password fields
3. Mobile-friendly web app
4. Master recognition through mnemonic words
5. Serial acts as 2\textsuperscript{nd} factor
Under the hood

• Master Password dictionary-based entropy measured as it is typed
  • No credit for blacklisted words
  • Mnemonic “Hashili” hash optionally displayed (10,000 variations)

• Master + page origin + serial fed to SCRYPT
  • Number of rounds inversely proportional to Master entropy
  • Make it very expensive to brute-force or make rainbow table

• Modify base64 output for length and character type
Example: Google.com

• Master Password = “I love fried twinkies” (entropy = 68.4 bits, hashili = “jero”)

• If no serial, feed into SCRYPT with salt “google.com” and do $2^{11}$ iterations (from entropy)

• Result in base64:
  //q/VO+095APfpV70Rpxk51myH/GXHWbvcoFTuwBlVQ=

• Replace all A a + / = with _ first B b C c with !, first D d E e with # (default special chars.)

• Output:
  __q_VO_095_PfpV70Rpxk51myH_GXHW!vcoFTuwBlVQ__
With serial

• Same Master, but now add serial “1”
• Now the salt fed into SCRYPT is “google.com1”
• Result after adding special characters:

M3wuwwPU1t#5kg9UmrsOpt!TWo5B_U9bEztQ4__iqJ8__

(without serial was: __q_VO_095_PfpV70Rpzxk51myH_GXHW!vcoFTuwBlVQ__)
Demo of SynthPass
Conclusions

• Synthesizers are better than Managers:
  • Always high strength passwords
  • No storage that might be compromised/corrupted/go out of sync
  • Free

• But (most of them) are worse than Managers:
  • Hard to change password
  • Clunky interface
  • No mobile

• SynthPass designed to combine the best of both kinds
SynthPass available at

• Chrome web store
• Firefox web store

• Mobile app at synthpass.com/app

• Next version may have encryption capability