A Modern Take on Passwords

Jim Fenton
@jimfenton

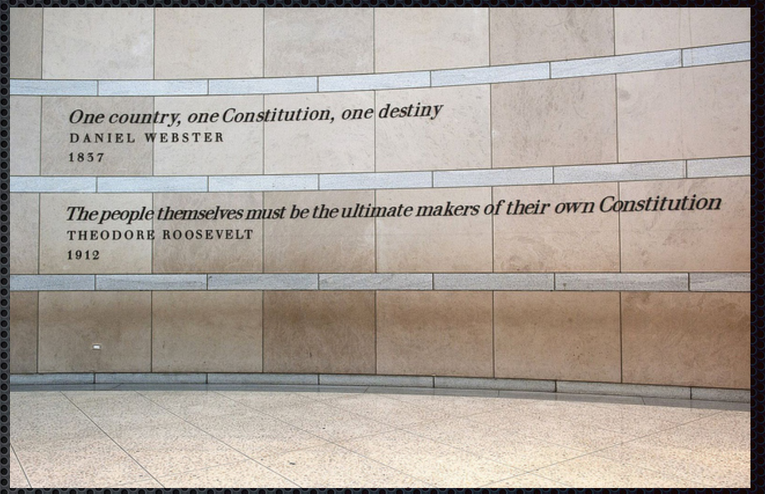
Just a little about me…

- Consultant (2013-present)
  - Authentication standards: NIST SP 800-63-3
  - IETF: REQUIRETLS email security proposal
- CSO at OneID (2011-2013)
  - Authentication startup
- Distinguished Engineer at Cisco (-2011)
  - Various things including DKIM email signatures

Disclaimer

- I’m a consultant for the US National Institute of Standards and Technology
  - Worked on the SP 800-63-3 update
  - Currently working on errata, guidance for US agencies
- Everything here is my own (hopefully informed) opinion
  - I don’t speak for NIST!
- Please contact NIST if you need an official answer

Guiding principles



- Emphasize user experience
  - People cheat when things are not user-friendly
- Have realistic security expectations
  - Many things need 2-factor authentication
- Burden the verifier rather than user wherever possible
- Don’t ask the user to do things that don’t significantly improve security
- Remember that the goal is to help real users authenticate, not just stop bad actors

Who are the Users?

- Everybody:
  - Non-English speakers
  - Homeless people
  - Disabled veterans
  - Hospital patients
  - Physicians
  - Elderly
  - Students
- Usability needs to consider all of these

Photo by Rob Curran on Unsplash

What's a password?

Passphrase

PIN

Something you know

Memorized Secret

Passcode

Attacks

- Online
  - Various types of guessing
- Offline
  - Attacks on the verifier
- Side channel
  - Shoulder surfing and more sophisticated attacks
- Both targeted (e.g., spearphishing) and bulk: very different goals

Online attacks

- Guessing the password
  - Brute force attacks
  - Password stuffing (passwords this user uses elsewhere)
- Common defenses
  - Throttling (more on this later)
  - Password reuse avoidance (education primarily)
  - Prohibition of very common passwords

Offline attacks

- Reversing password hashes (cracking)
  - More efficient with time: Moore's Law
  - Benefits from cryptocurrency mining technology, GPUs, etc.
- Defenses
  - Time- and memory-hard hash algorithms
  - Supplemental keyed hashing
  - Protecting hashes better!
- Generally harder to defend against than online

Side channels

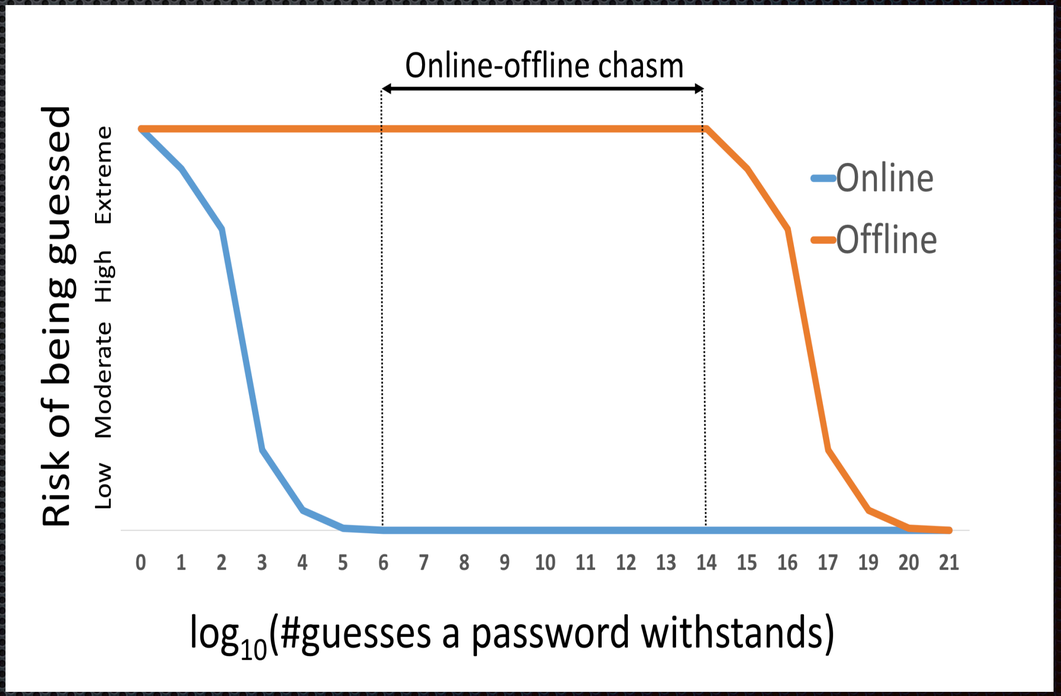
- Obtaining the password through leakage
  - Shoulder surfing
  - Key loggers and other malware
  - Acoustic, key wear, and similar analysis
  - Electromagnetic (TEMPEST), timing, and power drain analysis

Password length

- Increasing length is the most reliable way of strengthening a password
- So just require very long passwords?
  - No. Have a rationale for length requirements
  - Don’t drive users to Post-It® notes

What are you defending against?

- ~8 character passwords are effective against online attacks (with reasonable throttling)
- But it takes more than twice as many characters to provide similar protection against offline attacks

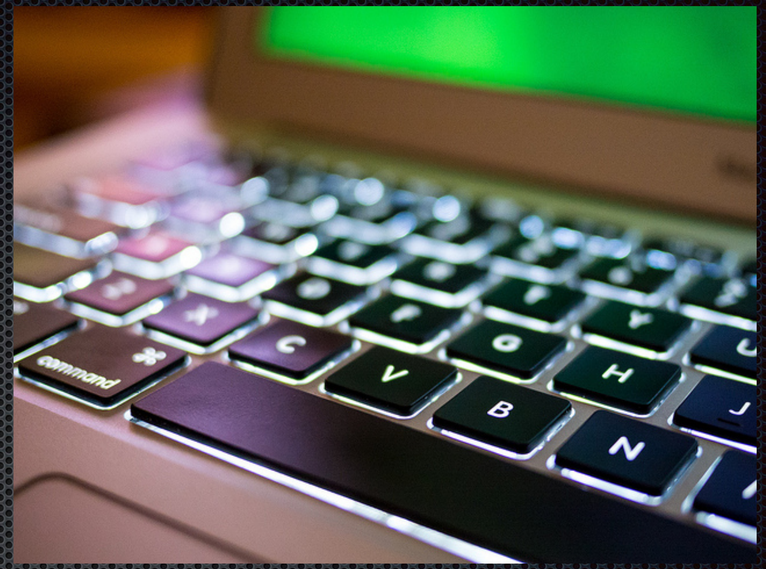


Florêncio, Dinei, Cormac Herley, and Paul C. van Oorschot. "An Administrator's Guide to Internet Password Research." *Usenix LISA*, November 2014. http://research.microsoft.com/apps/pubs/default.aspx?id=227130.

Maximum length

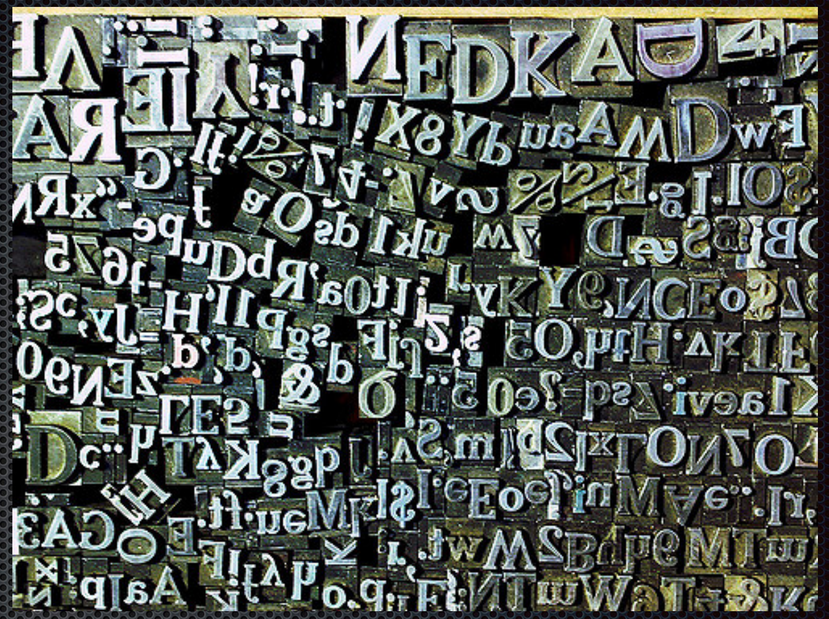
- Don’t limit users’ ability to use long (secure!) passwords!
- Suggest accepting 64 characters or more
- Rationale:
  - Give users maximum flexibility to choose a memorable pass phrase
  - 64 characters fit on many screens

Space characters



- Spaces are natural to type in passphrases: Allow them!
- Consider normalizing multiple consecutive spaces to one
  - UI concern: inadvertent typing multiple spaces is hard to see
  - Space characters themselves don’t add much entropy
  - (This is controversial)

Character set



- Give users maximum flexibility to choose passwords in their native language
  - Accept all printable ASCII characters
  - Accept Unicode, including emojis (1 “character”/code point) 😺
- Rationale:
  - Site-specific constraints on special characters have been a UX nightmare
  - Verifier needs to hash the entry anyway, so SQL injection shouldn’t be a concern

Hints and prompts

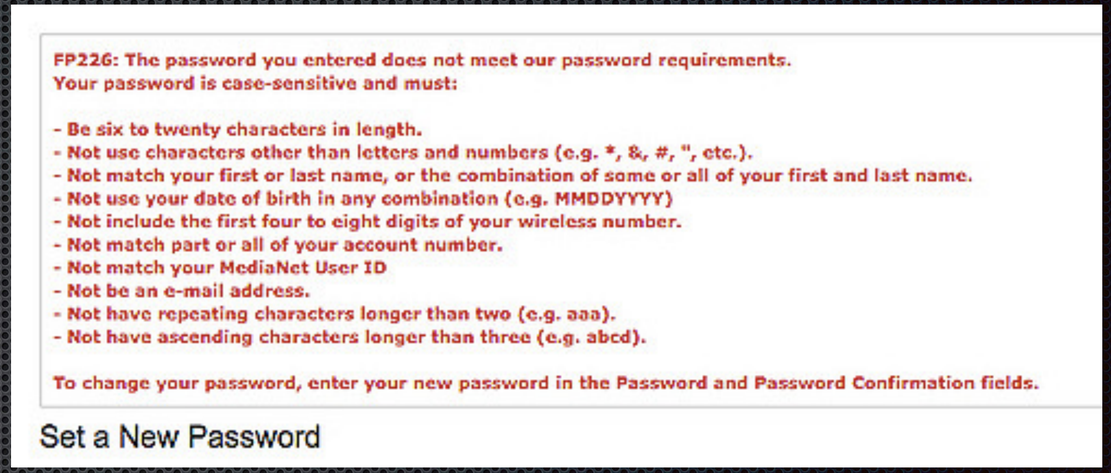
- tl;dr: Don’t do it!
- Hints (user-chosen)
  - Users sometimes choose hints like “Password is qwertyui”
  - Need to be stored in the clear or reversibly encrypted to be displayable to the user
- Prompts (site-chosen)
  - Typically take the form of “security questions”
  - Answers often shared between different services (e.g., first pet)

Throttling

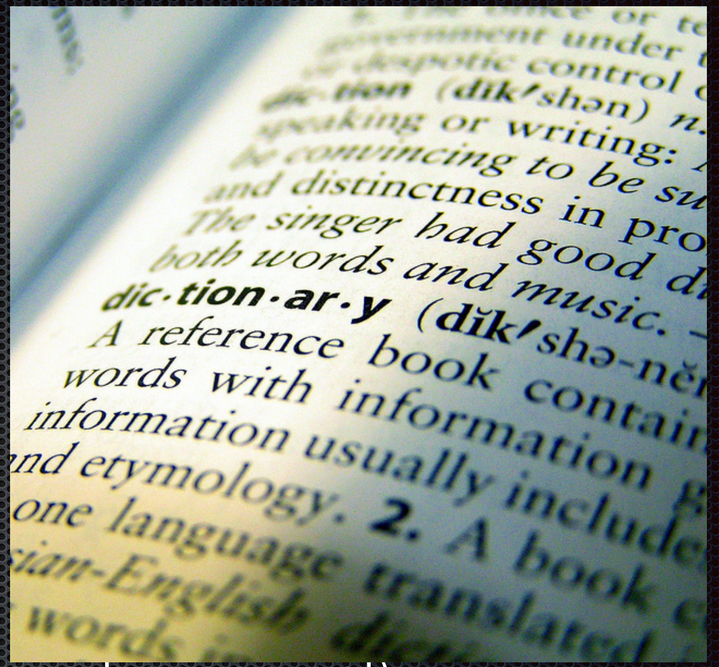
- Primary defense mechanism for online attacks
- Example: Limit failed authentication attempts to 100 in 30-day period per account
- Consider using CAPTCHAs, delays, or IP whitelists when approaching the limit
- Consider use of risk-based or adaptive techniques for throttling
- Don’t over-throttle: can result in

Composition rules

- Rules specifying what character classes must be in passwords
- Avoid using them:
  - UX nightmare
  - Don’t provide as much value as originally thought
  - May not be applicable in other languages
- Use a blocklist dictionary instead



Dictionaries: questions



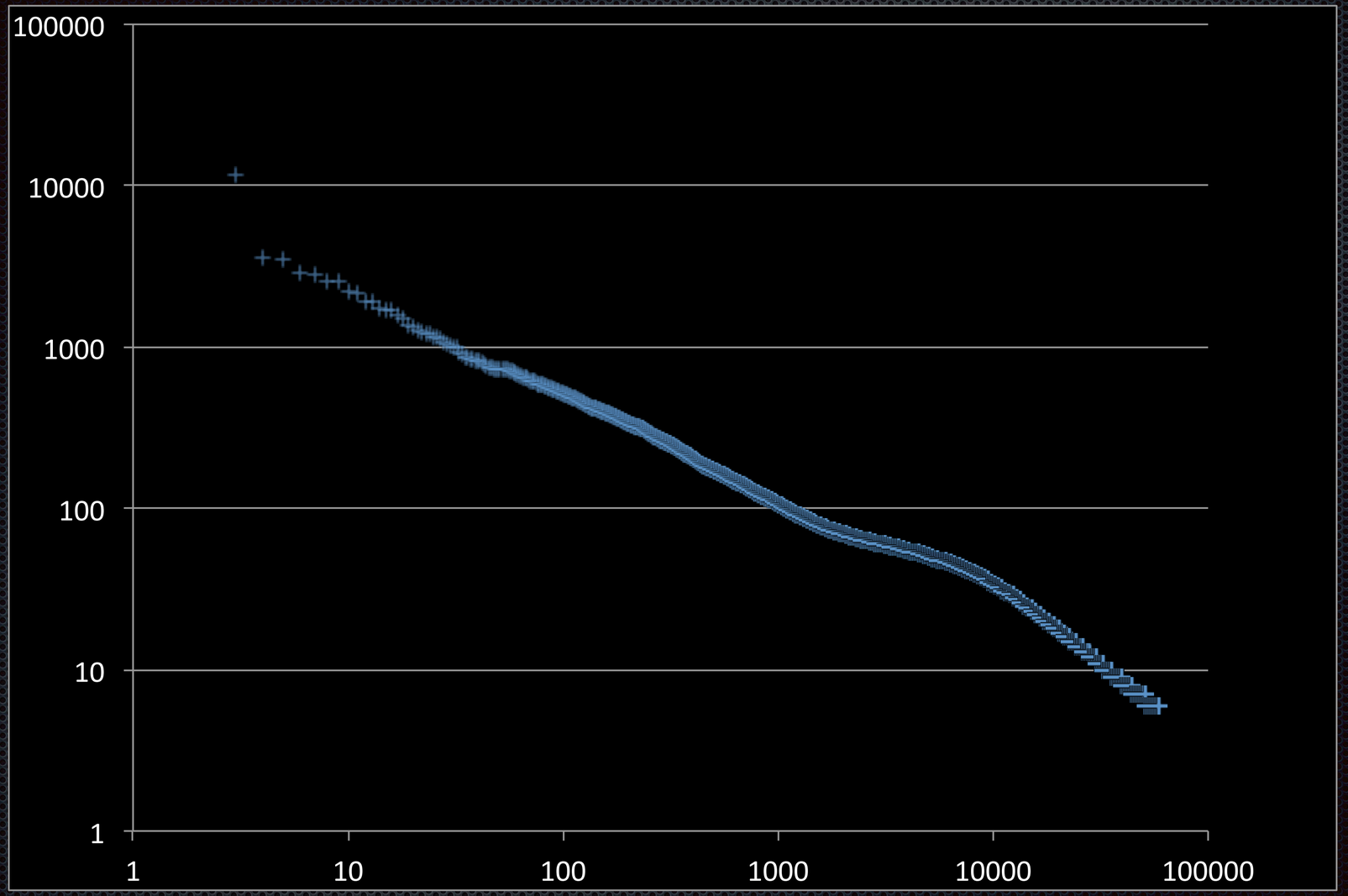
- How big should the dictionary be?
  - Too small: ineffective
  - Too big: bad user experience (like composition rules, but less transparent)
- Will users act predictibly when asked to pick a different password?
  - Users might just append something like 1 or !
  - If so, the dictionary is a great resource for offline cracking
  - Consider using a password strength indicator to coach users on subsequent picks

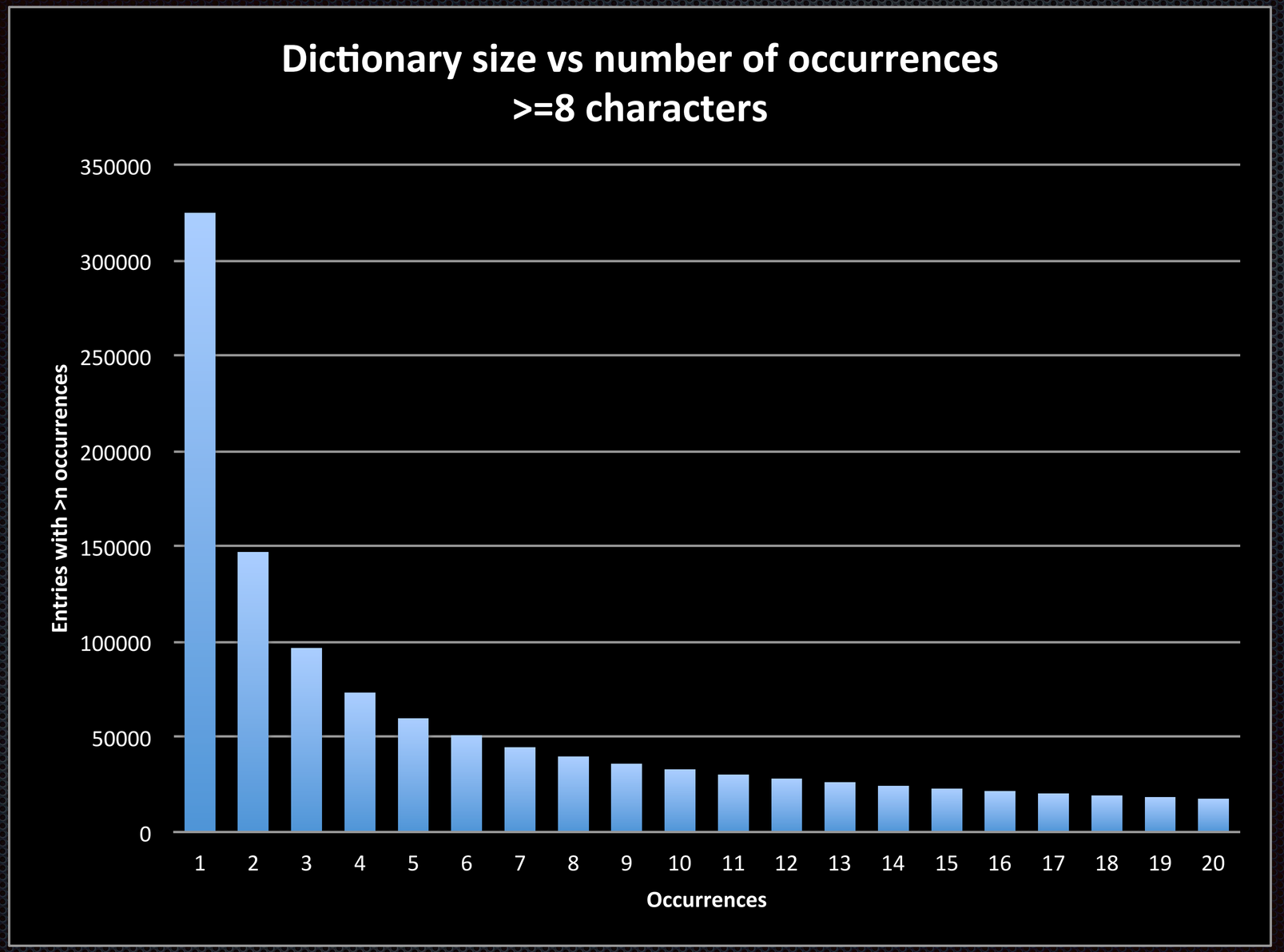
Dictionary investigation

- What would a good dictionary look like?
  - How big?
  - What's in it?
- Started with Burnett's list of 10M compromised passwords
  - Limited to >=8 characters
  - 4945022 entries, 3199670 distinct passwords


Dictionary Distribution
100000
10000
1000
100
10
1
password
15041988
27041987
motdepasse
11011994
06101990
03081974
welcome123
hellrais
travolta
bowhunter
18091968
gjkrjdybr
galatasa
sebring1
04112002
5858855abc
31011999
Vampire1
combined
31423142
stryker1
greencard
30031954
pointer1
actually
jackknife
asianpus
griffey2
pc7fddmh
yfcnz1994

Dictionary distribution (log-log scale)




Dictionary size vs number of occurrences
>=8 characters
Entries with >n occurrences
350000
300000
250000
200000
150000
100000
50000
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Occurrences

‘;--have i been pwned?

- Database of compromised accounts/passwords collected by Troy Hunt
- Huge: 6,474,030,172 accounts (as of 14 Feb 2019)
- Excellent API for securely checking passwords against compromises
- But is this too big? Will it frustrate users?
- https://haveibeenpwned.com

What if someone picks a bad password?

- If a user picks a password that's in the dictionary, this is a teaching opportunity 😀
- CMU has done some research on this [Habib 2017]
- Password strength meter might help user pick something stronger

Dictionaries: takeaways

- It’s pretty simple to build a reasonable dictionary
- Dictionary with size of ~100,000 entries is probably good - but need to test
- But watch out for that second password pick
  - BadPassword -> BadPassword1 ??? 😰

Displaying passwords

- Much of the time, users aren’t subject to shoulder-surfing attacks
- Consider offering option to display the password rather than dots or asterisks
  - But rehide after some period of time
- Displaying the password when not likely to be observed helps typing accuracy, and therefore improves user experience

Password expiration

- Don’t require users to change passwords arbitrarily (e.g., periodically)
- If users know their password will be only temporary:
  - They won’t invest the effort in choosing and learning a complex one
  - They’ll pick something similar to the old password
- But do require change if there is evidence of compromise
  - Have a way to do this, if/when needed.

Designing password verifiers

Hashing

- Goal: Make it hard for someone who compromises the verifier to learn the password(s)
- Simplistic approach:
  - Store sha256(password)
  - But: Attacker could try lots of passwords and see what matches
  - But also: Attacker can easily see if two users have same password

Salting

- Addresses uniqueness of hash for a given password
- At password establishment, choose a random value (“salt”)
- Store salt, sha256(password || salt)
- Foils look-up tables (or makes them very big), duplicate searches
- But: it’s still really fast. Attacker can just guess

Iterated hashing

- Goal: make guessing more expensive for the attacker
- Store salt, iteration count “n”, pbkdf2(password || salt, n). But:
  - pbkdf2 runs well in graphics processors, doesn’t require much memory
  - Benefits from technology developed for cryptocurrency mining
- Example:
  - pbkdf2_sha256$30000$Da4AnjGEyPCK$WjRjDzeJTaFzLzDWXV0av0Z5jE7o8mDFEfP9cPvQ9BQ=
  - Algorithm $ Iteration count $ salt (base64) $ hash (base64)

Time and memory hardening

- Good algorithms for password hashing are:
  - Slow - requires processor resources (“time hard”)
  - Memory consumptive - requires memory resources (“memory hard”)
- Attackers have access to great CPU resources, specialized hardware
- Popular algorithm: bcrypt

Case study: Adobe®

- Reported October, 2013
- 130,325,129 records containing:
  - Email address
  - Encrypted password (not salted)
  - Password hint (not encrypted)
- 56,044,956 distinct encrypted passwords (many duplicates)

Adobe

- Problems:
  - Passwords used by multiple people have the same stored value
  - Correlation of hints is possible — this is actually a fun game!
  - Email address facilitates credential stuffing on other services
- Successes:
  - Encryption key apparently not breached (but who can be sure?)
  - Cracking of hashes not possible because of key

The Adobe game

0agIJWqXa2Y= (697 records, 276 hints, 152 distinct hints)

chugalug
same
beverage
kitty
drink
uncle drunk -
chien
Glenlivet
drunk
booze
whiskey
horse
tape
sweet
andrews son
p
male cat
liqour
What's my dog's name?
cat
whos ur dawg
bunny
Border Collie
your favourite liquor
doggy
dog
tapes
reuse

mu dog
normal
sticky & plaid
black label
favorite drink
geknipst
my dog
puppy!
alcohol
aged 25 years
whisky
school
attaccare
friendf
ehhhhhhhh
dimple
dogs
scortcher
pets name
ura
Irish Setter
First Dog
fave horse
golden
normal one
carpetCleaner
favorite liquor
john's password

mon chien
hi priced liquor
gina
minou
tape?
my dogs name
what i drink
usual
its the same ol brand
boycatname
same as always
lijm
Better than Cats
plakband
Dog On Habbo
on the rocks
college
bant
Dewers
Name of Dog
drink and tape
Fav. Disney 1
?????????
Cutts nickname for me
scotland
tacataca
ZK
a good drink

favourite cat
publish
cat name
scotc
sc
SVR
cats first name
little one
montreal
????
drotch
liquor
cat's name
yellow snake
me
short
Dog's name?
mom's dog
lenrelax
partner
puppy
FPIS Favorite
favorite drink haha
loly
land
down in my belly
highland vid tape mfr
woof

first horse
estimac?o
c?o
speyside
malt
Pet
cats name
regular
dizzle
gatto
libation w/o number
johnny walker
persian
alex
name of school in Hawthorn
Tony & Lou Drink
new dog
buro
favourite drink
college australia
pyranees
zelda
pet's name
first first
single malt
yummy beverage
nom du chien

nationality
the usual
Favorite Dog
DTH favorite beverage
favorite team
log in pass
trunek
albert minus wendy
Hund
Same drink
cinta magica
Horse name
Omi

Adobe — Lessons

- Lack of salt causes one breached password to impact perhaps hundreds of accounts
- Password hints are evil
- Often easier to protect one key stored separately than a large database

“Security” questions

- Also known as Knowledge-Based Authentication (KBA)
- Aren’t these just passwords with hints?
- *Something you know*, so KBA+password isn’t 2-factor
- Low entropy, likely to be reused on multiple sites
- Can’t be hashed if fuzzy matching is needed

First pet?

Case study: Ashley Madison

- Data breach, July 2015
- Included cleartext answers to KBA questions
- Limited choice of questions, e.g., “What high school did you attend?"
- Popular answer, Central High School, was represented in many ways: central hi, Central HS, Central, etc.

Questions?

Bibliography

- [Herley 2012] Herley, C., and P. Van Oorschot. 2012. “A Research Agenda Acknowledging the Persistence of Passwords.” IEEE Security & Privacy Magazine 10 (1): 28–36. https://doi.org/10.1109/MSP.2011.150.
- [Florencio 2014] Florêncio, Dinei, Cormac Herley, and Paul C. van Oorschot. 2014. “An Administrator’s Guide to Internet Password Research.” Usenix LISA, November. http://research.microsoft.com/apps/pubs/default.aspx?id=227130.
- [Grassi 2017] Grassi, Paul A, James L Fenton, Elaine M Newton, Ray A Perlner, Andrew R Regenscheid, William E Burr, Justin P Richer, et al. 2017. “Digital Identity Guidelines: Authentication and Lifecycle Management.” NIST SP 800-63b. Gaithersburg, MD: National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-63b.
- [Habib 2017] Habib, Hana, Jessica Colnago, William Melicher, Blase Ur, Sean Segreti, Lujo Bauer, Nicolas Christin, and Lorrie Cranor. 2017. “Password Creation in the Presence of Blacklists.” In Proceedings 2017 Workshop on Usable Security. San Diego, CA: Internet Society. https://doi.org/10.14722/usec.2017.23043.