Passwords/multi-factor authentications

Mainack Mondal

CS 60081
Autumn 2022

Roadmap

- Passwords/multi factor authentications
- Usability for security developers
- Temporal aspect of privacy
- Online tracking
- Privacy notices/dark patterns
- Privacy policies

Roadmap

- **Passwords/multi factor authentications**
- Usability for security developers
- Online tracking
- Temporal aspect of privacy
- Privacy notices/dark patterns
- Privacy policies

Good authentication system should be …

- User friendly
  - Memorable, scalable, physically no/little effort, easy to learn, infrequent errors, easy to recover from fault
- Ease of implementation
  - Accessible, negligible cost per user, server deployable, browser deployable, free
- Protection against
  - Targeted impersonation, throttled guessing, unthrottled guessing

Passwords

Word/phrase only known by user. User authenticates herself by providing passwords to the server which then verifies that it is the correct one

Password expectation

Roger Buffle Jr. supplies his father with yet another computer password.

Password creation

SELECT A NEW PASSWORD... PLEASE INCLUDE LETTERS, NUMBERS, RUNES, HIEROGLYPHICS, & SMOKE SIGNALS...

Password reality

PASSWORD
abc123
WEAK!
BUT 'ABC123' ARE THE ONLY LETTERS AND NUMBERS I CAN REMEMBER.
SLANE co.nz
Easy to remember = easy to guess

Why passwords?

- Password related issues are identified as one of the top ten security related problems

- Password == often only barrier an attack needs to cross
  - Frequently misused
  - Hard to create and memorize passwords

Most popular passwords of 2020

- 123456
- 123456789
- qwerty
- password
- 1234567
- 12345678
- 12345
- iloveyou
- 111111
- 123123

Most popular passwords of 2022

- 123456
- 123456789
- qwerty
- password
- 12345
- qwerty123
- 1q2w3e
- 12345678
- 111111
- 1234567890

The problem: Users are not the enemy

- Users are not the enemy (Adams and Sasse, 1999)
  - https://discovery.ucl.ac.uk/20247/2/CACM%20FINAL.pdf
  - Studied factors impacting compliance with password policies at 2 organizations
    - Many people wrote their passwords down
    - Struggling to cope with having to frequently change and remember – results in simpler passwords
    - Users don't understand what makes a password secure
  - Part of the problem is users don't know security risks and rationale for security procedures

The method

- Web-based questionnaire
    - Focused on password related behaviors
    - Then 30 semi-structured in-depth interviews
- Analysis technique
    - Grounded theory
    - Build concepts based on data

Problems of passwords

- 1) multiple passwords
  - Hard to remember
- 2) password content
  - Created password based on what they know
- 3) perceived compatibility with work practices
  - Shared password within a team vs. individual passwords
- 4) users’ perceptions of organizational security and information sensitivity

Organizational problem

Communication between security departments and users is therefore often restricted to “ticking off” users caught circumventing the rules … Users have to be treated as partners in the endeavor to secure an organization’s systems, not as the enemy within. System security is one of the last areas in IT in which user-centered design and user training are not regarded as essential; this has to change.

A good solution: password managers

Do people reuse passwords?

- Password Management Strategies for Online Accounts
  - https://cups.cs.cmu.edu/soups/2006/proceedings/p44_gaw.pdf
  - SOUPS’2006
  - Material from the slides by authors

Do people reuse passwords?

- Part laboratory exercise and part survey
- Part 1
    - Select from 139 websites
    - Login to each website
    - Self-report summary statistics (#passwords they used for all websites)
- Part 2
    - List other websites used personally
    - Re-report summary statistics

Password reuse

- Unique passwords: *M* = 3.31, *SD* = 1.76 (*n* = 49)
- Passwords reuse rate: *M* = 3.18, *SD* = 2.71

Password Reuse

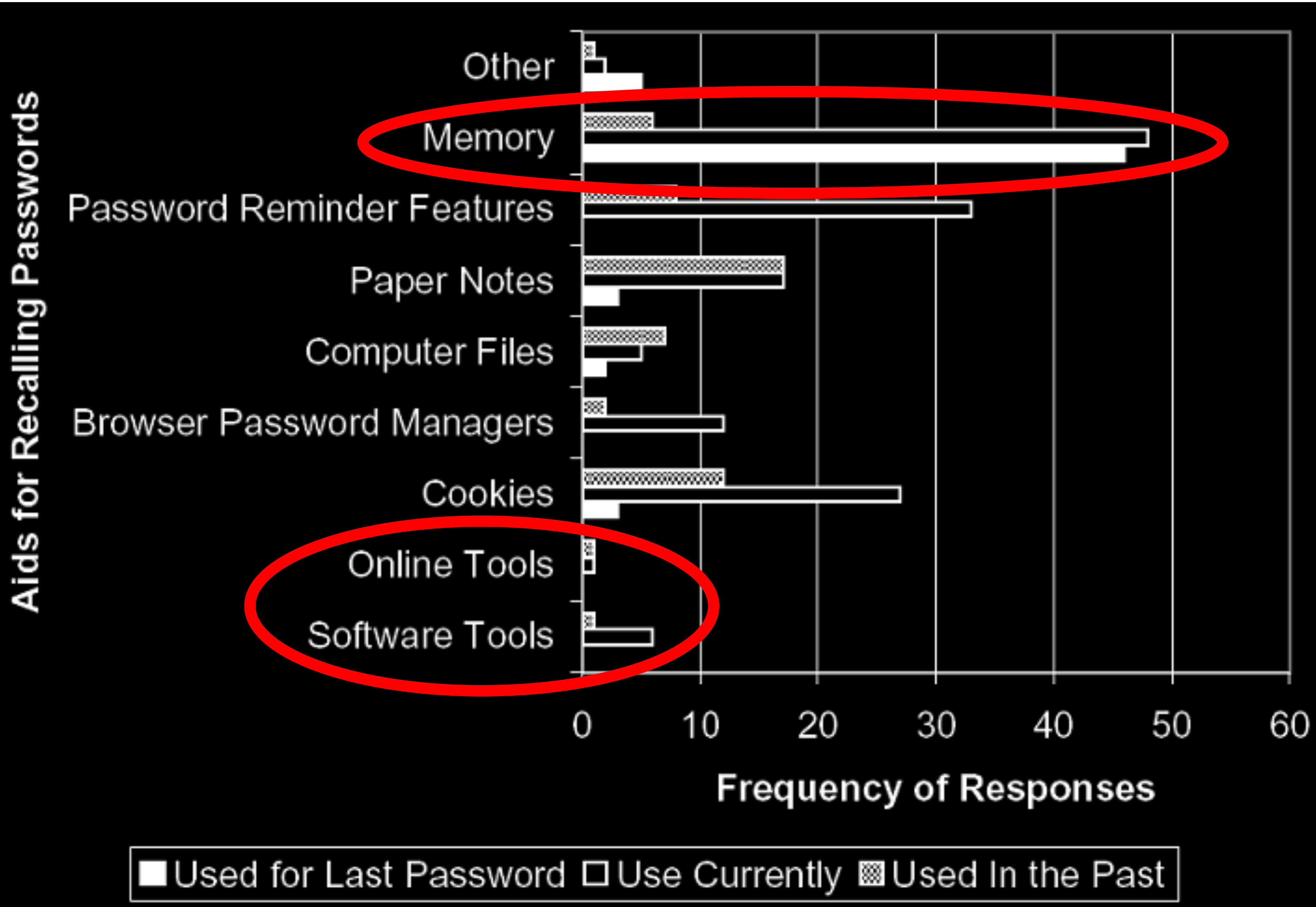
People will **reuse** passwords more as they acquire more accounts

Reasons for Reuse: Method

115 question survey

(n = 58)

- Demographic information
- Explanations of password reuse/avoidance
- Descriptions of password creation/storage
- Descriptions of password management


Aids for Recalling Passwords
Other
Memory
Password Reminder Features
Paper Notes
Computer Files
Browser Password Managers
Cookies
Online Tools
Software Tools
0
10
20
30
40
50
60
Frequency of Responses
■ Used for Last Password
□ Use Currently
▨ Used In the Past

Reasons for Reuse: Results

Why use the same password?

It is easier to remember (35)

People rely on their memory rather than store passwords

So, how do you crack passwords?

- Slides partially from : https://www.cs.purdue.edu/homes/ninghui/courses/526_Fall14/handouts/14_526_topic07.ppt

Variants of Passwords

- Password
- Passphrase
  - a sequence of words or other text used for similar purpose as password
- Passcode
- Personal identification number (PIN)

Guessing Attacks: Two Factors for Password Strength

- The average number of guesses the attacker must make to find the correct password
    - determined by how unpredictable the password is, including how long the password is, what set of symbols it is drawn from, and how it is created.
- The ease with which an attacker can check the validity of a guessed password
    - determined by how the password is stored, how the checking is done, and any limitation on trying passwords

Password guessing: Setup

- Passwords are generally hashed before storage
  - Why? Attack model
  - Called “*password cracking*”

- Password file is publicly readable
  - Unix, set of password hashes from data breaches

- Any user can try “offline dictionary attack”

Password guessing: Dictionary attack

- Dictionary attack
  - User looks into a password dictionary
  - Computes hash for every word in dictionary
  - Match with the hash they see
- Password dictionary
  - ~1m entries (names, pet names, ordinary words)
- Efficiency
  - With 10 guesses/s , you need 100,000 sec = 14 hours
  - For brute force on a 6-character password with uppercase and lowercase, 32 punctuation, digits = 1,093 years

Guessing attack protection: salting

- Salt = random string
    - Password is stored as [User1, Salt1 , HASH(User1, Salt1)]
- Without salt
    - Same password, always same hash
    - Precompute the hash
- With salt (say 12 bit salt)
    - One password is hashed 2^{12} different ways
    - Cannot precompute hash file
    - For each salt in file needs to compute dictionary
    - Hashes like **bcrypt** are used which are slow to compute

But how to know if your users' password is good?

depends on attack model

Password Entropy

- The entropy bits of a password, i.e., the information entropy of a password, measured in bits, is
    - The base-2 logarithm of the number of guesses needed to find the password with certainty
    - A password with, say, 42 bits of strength calculated in this way would be as strong as a string of 42 bits chosen randomly.
    - Adding one bit of entropy to a password doubles the number of guesses required.
- Aka. Guess entropy

Estimating Password Entropy

- People are notoriously remiss at achieving sufficient entropy to produce satisfactory passwords.
- NIST suggests the following scheme to estimate the entropy of human-generated passwords:
  - the entropy of the first character is four bits;
  - the entropy of the next seven characters are two bits per character;
  - the ninth through the twentieth character has 1.5 bits of entropy per character;
  - characters 21 and above have one bit of entropy per character.
- This would imply that an eight-character human-selected password has about 18 bits of entropy.

Towards Better Measurement of Password Entropy

- NIST suggestion fails to consider usage of different category of characters:
    - Lower-case letters, digits, upper-case letters, special symbols
- Orders also matter:
    - “Password123!” should have different entropy from “ao3swPd!2s1r”
- State of art is to use variable-order markov chains to model probability of different strings as passwords
    - “A Study of Probabilistic Password Models” by Ma, Yang, Luo, Li in IEEE SSP 2014.
- Fundamental challenge: there are different attack strategies out there, which try passwords with different ordering

Practical approach: Hashcat

Practical approach: John the ripper (JtR)

Mechanisms to Avoid Weak Passwords

- Allow long passphrases, forbid short passwords
- Randomly generate passwords where appropriate
  - Though probably inappropriate for most scenarios
- Give user suggestions/guidelines in choosing passwords
  - e.g., think of a sentence and select letters from it, “It’s 12 noon and I am hungry” => “I’S12&IAH”
  - Using both letter, numbers, and special characters
- Check the quality of user-selected passwords
  - Use a number of rules of thumb; run dictionary attack tools
  - Evaluate strength of a password and explain the weaknesses

Password creation rules

- NIST in 2003 invented rules of how to create *good* passwords
  - letters, numbers, and symbols
  - changing every 90 days

How to enforce it? Strength meters

- Paper: "How does your password measure up? The effect of strength meters on password creation.", Usenix 2012
- RQ
    - What kinds of meters are being used by websites right now?
    - What are “good” measures of password quality?
    - How do different meter designs impact the passwords created? If so, which meters perform best?

RQ 1: What kinds of meters are being used by websites right now?

- How would you do it?

RQ 1: What kinds of meters are being used by websites right now?

- Reviewed login pages of Alexa top 100 most popular websites
  - 96 allowed a login
  - 70 gave some type of password feedback
  - Common types of meters
    - Bar-like (50%)
    - Checkmark or X system (41.3%) □
    - Text indicating problems (21.2%)

RQ 2: Good measures of password quality

- How would you solve it?

RQ 2: Good measures of password quality

- Look at earlier scientific literature to *borrow* ideas
- Basic16
    - Password contains >= 16 characters
- Comprehensive8
    - at least eight characters, uppercase letter, lowercase letter, a digit, and a symbol.
    - Not in a wordlist of common passwords

RQ3: How do different meter designs impact the passwords created?

- Online survey study using Amazon Mechanical Turk (AMT)
  - Shown15 different types of password meter
  - 2931 participants

- Study phases
  - Setup a password
  - 2 days later, log in using the original password

Meters tried

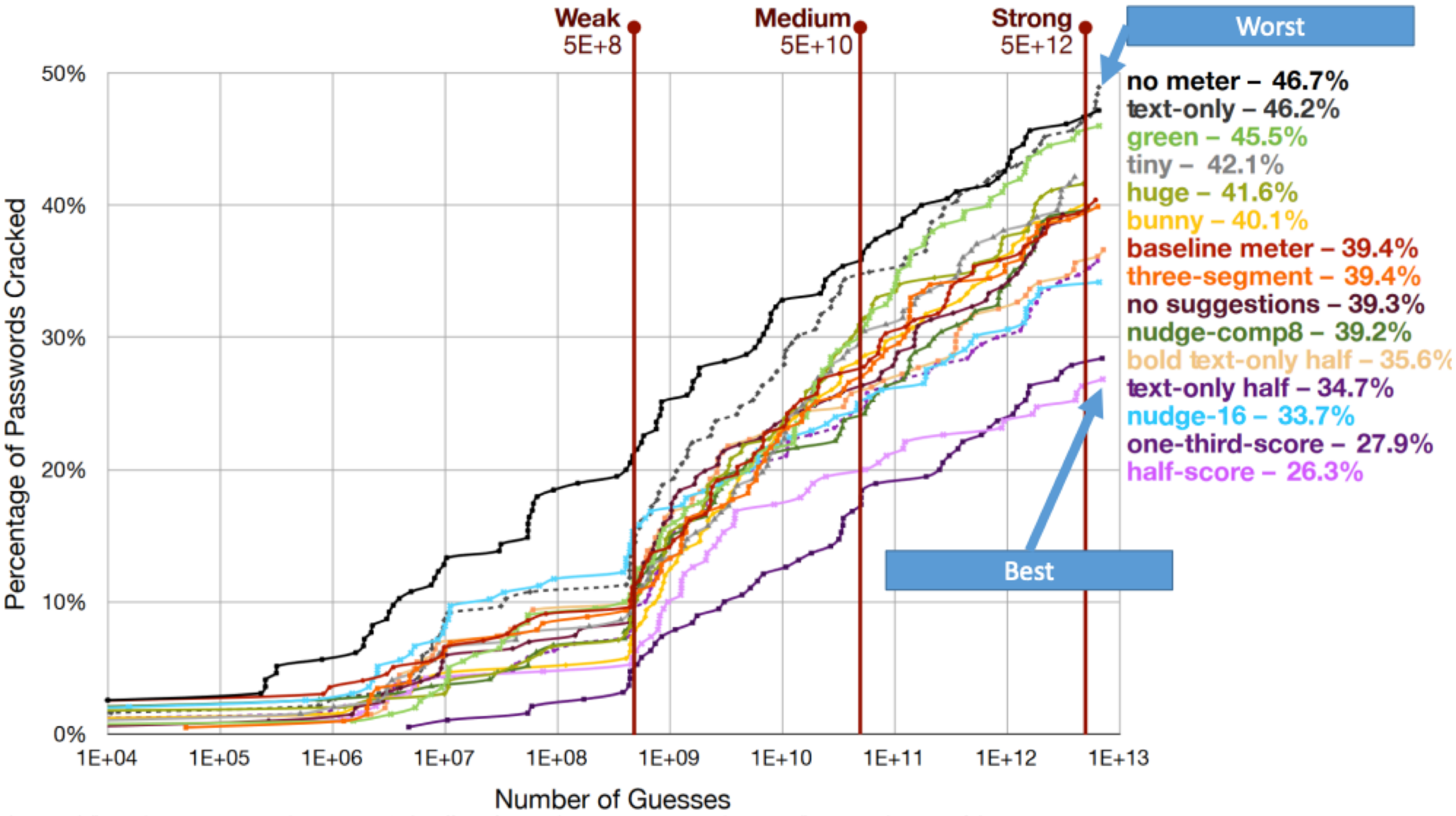
- Control: No meter
- Appearance variations: 3 segment, always green, tiny, huge ..
- Scoring: half-score (always half full), nudge-16, nudge-comp8
- Variation: Running bunny instead of a bar

Anything left in the design?

Anything left in the design?

- What is the attack model?

Results


Weak
5E+8
Medium
5E+10
Strong
5E+12
Worst
Best
50%
40%
30%
20%
10%
0%
Percentage of Passwords Cracked
1E+04
1E+05
1E+06
1E+07
1E+08
1E+09
1E+10
1E+11
1E+12
1E+13
Number of Guesses
no meter – 46.7%
text-only – 46.2%
green – 45.5%
tiny – 42.1%
huge – 41.6%
bunny – 40.1%
baseline meter – 39.4%
three-segment – 39.4%
no suggestions – 39.3%
nudge-comp8 – 39.2%
bold text-only half – 35.6%
text-only half – 34.7%
nudge-16 – 33.7%
one-third-score – 27.9%
half-score – 26.3%

Question

- Is there any shortcoming in the design?
    - Validity point of view?

Two factor authentication

- Something you know
  - Your password
- Something you have
  - Your device

Question: Is getting an OTP more secure than passwords?

Question: Is getting an OTP more secure than passwords?

- User friendly
  - Memorable, scalable, physically no/little effort, easy to learn, infrequent errors, easy to recover from fault
- Ease of implementation
  - Accessible, negligible cost per user, server deployable, browser deployable, free
- Protection against
  - Targeted impersonation, throttled guessing, unthrottled guessing

Question: Is getting an OTP more secure than passwords?

- User friendly
  - Memorable, scalable, physically no/little effort, easy to learn, infrequent errors, easy to recover from fault
- Ease of implementation
  - Accessible, negligible cost per user, server deployable, browser deployable, free
- Protection against
  - Targeted impersonation, throttled guessing, unthrottled guessing

Question: Is getting an OTP more secure than passwords?

- User friendly
  - Memorable, scalable, physically no/little effort, easy to learn, infrequent errors, easy to recover from fault
- Ease of implementation
  - Accessible, negligible cost per user, server deployable, browser deployable, free
- Protection against
  - Targeted impersonation, throttled guessing, unthrottled guessing

Roadmap

- Passwords/multi factor authentications
- **Usability for security developers**
- Online tracking
- Privacy notices/dark patterns