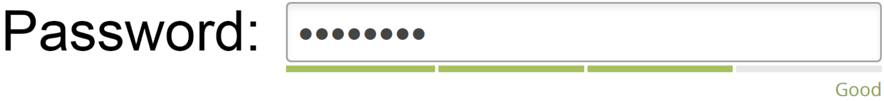
NDSS 2014 Presentation, Feb 25, 2014

# From *Very Weak* to *Very Strong*: Analyzing Password-Strength Meters

Xavier de Carné de Carnavalet Mohammad Mannan

Concordia University, Montreal, Canada

# Password-strength meter/checker

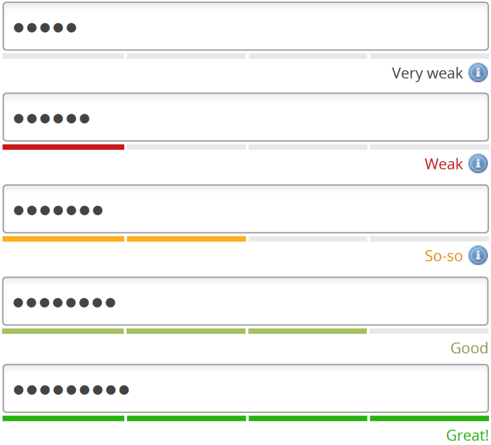


Password: ●●●●●●●●

Good

# What is this work about?

We analyzed why is this:



●●●●●

Very weak

●●●●●●

Weak

●●●●●●●

So-so

●●●●●●●●

Good

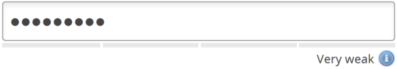
●●●●●●●●●

Great!

# What is this work about?

And why is that (same password):

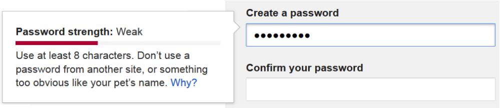
Dropbox



●●●●●●●●●

Very weak

Google



**Password strength:** Weak

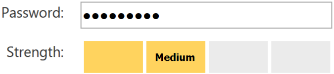
Use at least 8 characters. Don't use a password from another site, or something too obvious like your pet's name. Why?

**Create a password**

●●●●●●●●●

**Confirm your password**

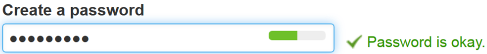
Microsoft



Password: ●●●●●●●●●

Strength: **Medium**

Twitter

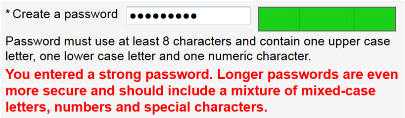


**Create a password**

●●●●●●●●●

✓ Password is okay.

FedEx



* Create a password ●●●●●●●●●

Password must use at least 8 characters and contain one upper case letter, one lower case letter and one numeric character.

**You entered a strong password. Longer passwords are even more secure and should include a mixture of mixed-case letters, numbers and special characters.**

# Our motivations

1. Recent studies: meters really guide users to choose better passwords [Ur *et al.*, USENIX Security'12] and [Egelman *et al.*, CHI'13]
2. Deployed meters impact hundreds of millions of users
3. Built by up-to-billion-dollar IT companies
4. They don't seem reliable...

# Tested 11 web services/applications

Dropbox
Google™
Drupal™
YAHOO!®
ebay™
twitter
PayPal™
skype™
FedEx®
Microsoft

# Analysis setup (1/3)

1. 11 dictionaries: 3,895,247 unique passwords

# Analysis setup (1/3)

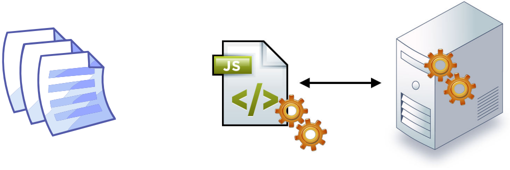
1. 11 dictionaries: 3,895,247 unique passwords
2. Top500, cracking tools (e.g., JtR) worm dictionaries, database leaks (e.g., RockYou)

# Analysis setup (1/3)

1. 11 dictionaries: 3,895,247 unique passwords
2. Top500, cracking tools (e.g., JtR) worm dictionaries, database leaks (e.g., RockYou)
3. Mangling & leet transformations

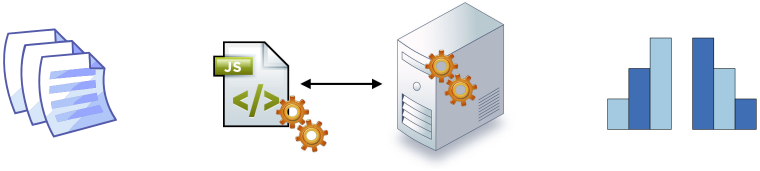
   *password* → *Password1+* or *p@5$w0rd*

# Analysis setup (2/3)



1. Understanding of functionalities (involve some RE)
2. JavaScript (whitebox) and/or server-side (blackbox)
3. 52+ million tests

# Analysis setup (3/3)



1. Analyze results
2. Understand checkers profile
3. Find common weaknesses

# In theory

Designing PSMs is non-trivial:

- No straightforward academic literature to follow
- Failure of NIST recommendations
- How to deal with password leaks, cultural references?

# In practice

- Custom “entropy” based on:
    - Perceived complexity
    - Password length
    - Number of charsets used
    - Known patterns
    - Comparison with dictionary of common passwords (blacklist)
- More entropy $\simeq$ more secure password
- Everyone invents their own algorithm

# Meters heterogeneity

1. Each meter reacts differently to our dictionaries
2. Strength results vary widely from one to another

Example: *Password1*

- Obvious, Very weak, Weak (x3), Poor, Moderate (blacklisted), Medium (x2), Strong (x3), Very strong
- By Microsoft itself (3 versions): strong, weak and medium!

3. Some simple dictionaries score significantly higher than others

# Stringency bypass

- Simple mangling rules/leet transformations allow bypassing password requirements

Example: Consider {Top500, C&A, Cfkr and JtR}

How many passwords are medium or better?

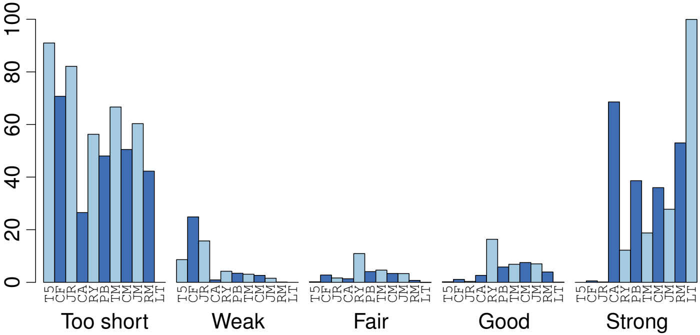
| **Web service** | **Regular** | **Mangled** |
|---|---:|---:|
| Skype | 10.5% | 78% |
| Google | 0.002% | 26.8% |

# Password policies

1. Password policies not often explicitly stated
2. Rules for measuring strength unexplained to users
3. Differences in policies:
   - Very stringent: assign strengths only for 3+ charsets (FedEx)
   - Promotion of single-charset passphrases (Dropbox)
4. Google and Yahoo!, lots of personal info, but lenient policy...

# Google checker: some results

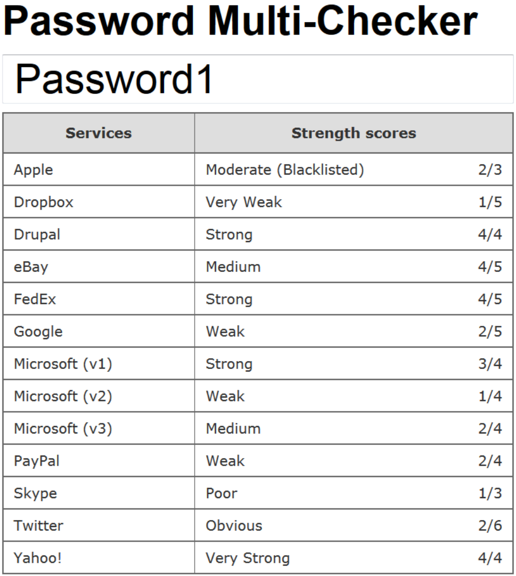
Password strength distribution:



Inconsistencies:

1. *testtest* is weak
2. *testtest0* is strong
3. *testtest1* is fair
4. *testtest2* is good
5. *testtest3* is strong...
6. Strength is time-dependent

# One checker to rule them all



## Password Multi-Checker

Password1

| Services | Strength scores | |
|---|---|---|
| Apple | Moderate (Blacklisted) | 2/3 |
| Dropbox | Very Weak | 1/5 |
| Drupal | Strong | 4/4 |
| eBay | Medium | 4/5 |
| FedEx | Strong | 4/5 |
| Google | Weak | 2/5 |
| Microsoft (v1) | Strong | 3/4 |
| Microsoft (v2) | Weak | 1/4 |
| Microsoft (v3) | Medium | 2/4 |
| PayPal | Weak | 2/4 |
| Skype | Poor | 1/3 |
| Twitter | Obvious | 2/6 |
| Yahoo! | Very Strong | 4/4 |

# Summary (1/2)

Facts:

- Passwords are not going to disappear anytime soon
- Users will continue to choose weak passwords

Current solutions:

- Stringent policies (user resentment?)
- Influence users in choosing better passwords, *willingly*
  - Provide feedback on the quality of chosen passwords
  - Should be consistent and avoid confusion

# Summary (2/2)

Reality:

1. Commonly-used meters are highly inconsistent
2. Fail to provide coherent feedback, sometimes blatantly misleading
3. Often have very ad-hoc design
4. Simple transformations not taken into account

# What can be done?

1. Common API to reduce confusion (e.g., Dropbox with *zxcvbn*)
2. Real-time cracking with state-of-the art techniques to assess passwords?
3. Passphrases (be careful at simple structures)
4. Password popularity, Markov models, PCFG, semantic?

# Thanks

To recap:

1. Meters less robust than expected from such large companies
2. Companies should stop misleading users
3. Opportunities for academic research

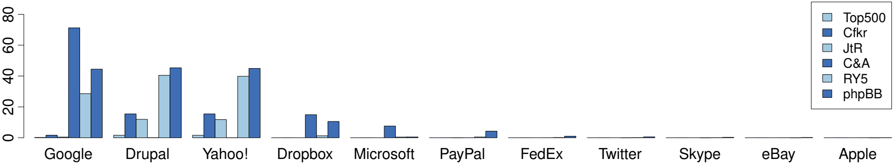
Contact: x_decarn@ciise.concordia.ca
Project URL: http://goo.gl/0E5Ieu

O,u3$T1()|\|5?

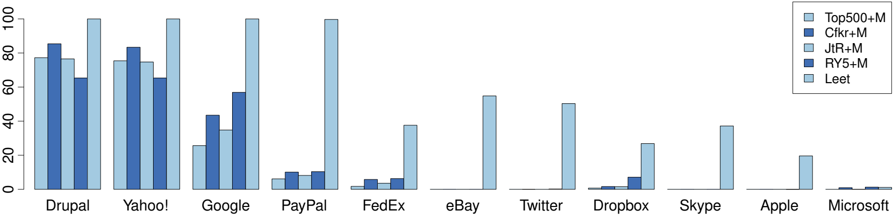
# Additional slides

# Percentage of dic. assigned "good" or +

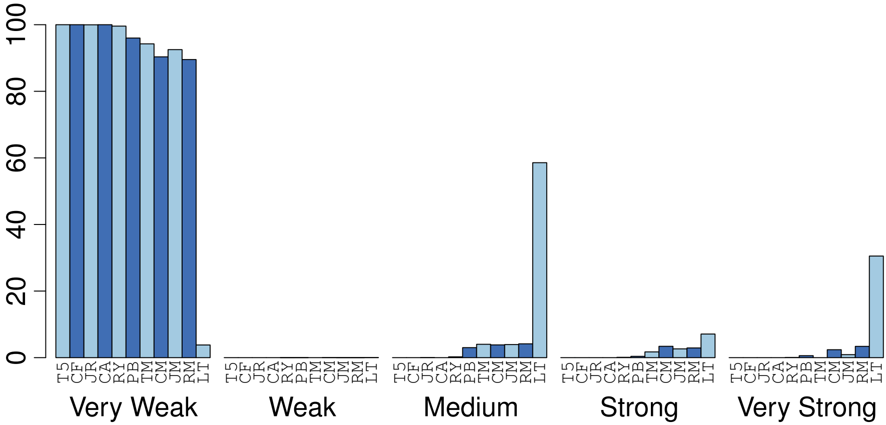
Base dictionaries:



"Advanced" dictionaries:



# FedEx: Password strength distribution



# FedEx: Password strength distribution

Very weak? Fine...

# FedEx: Targeted dictionary

Refined mangling rules:

1. capitalize, append a digit and a symbol
2. capitalize, append a symbol and a digit
3. capitalize, append a symbol and two digits
4. capitalize, append a symbol and a digit, and prefix with a digit

Gives 121,792 words from {Top500, JtR, Cfkr}

1. 60.9% is now very strong
2. 9.0% is strong
3. 29.7% is medium
4. 0.4% is very weak