Passwords (2)

Tom Ristenpart
CS 6431

The game plan

- Refresh from Tuesday
- Measuring password distributions:
  - Florencio & Herley (client-side measurement)
  - Bonneau (server-side measurement)
  - Understanding password strength metrics
- PCFGs and neural network models of password distributions
  - Weir et al. (PCFGs)
  - Melicher et al. (neural networks)

The research landscape since 1979…

- **Understanding user password selection**
  - Measuring password strength [see citations in Bonneau paper], [Li, Han `14], [CMU papers]
  - Measuring password reuse
- **Usability**
  - Strength meters, requirements, etc. [Komanduri et al. ‘11] [Dell’Amico, Filippone ‘15] [Wheeler ‘16] [Melicher et al. ‘16]
  - Password expiration [Zhang et al. ‘12]
  - Typo-tolerance [Chatterjee et al. `16]
- **Password transmission, login logic**
  - Single sign-on (SSO) technologies
  - Password-based authenticated key exchange [Bellovin, Merritt ‘92]
- **Password hashing**
  - New algorithms [PKCS standards], [Percival ’09], [Biryukov, Khovratovich ‘15]
  - Proofs [Wagner, Goldberg ‘00] [Bellare, Ristenpart, Tessaro ‘12]
- **Improving offline brute-force attacks**
  - Time-space trade-offs (rainbow tables) [Hellman ’80], [Oeschlin ‘03], [Narayanan, Shmatikov ‘05]
  - Better dictionaries [JohntheRipper], [Weir et al. ‘09], [Ma et al. ‘14]
- **Password managers**
  - Decoy-based [Bojinov et al. ’10], [Chatterjee et al. ‘15]
  - Breaking password managers [Li et al. ‘14] [Silver et al. ’15]
  - Stateless password managers [Ross et al. ’05]

Florencio & Herley 2007 study

- Instrument Windows Live toolbar
  - – 544,960 clients opted-in to study
- Captured passwords typed into browser
  - – Hashed and stored locally
  - – Sent report to server about (quantized) password strength, associated URL, etc.

Florencio & Herley 2007 study

- Avg user:
  - Has 6.5 passwords, each used at 3.9 different sites
  - Has 25 accounts requiring passwords
  - Types 8 passwords per day
  - Selects 40.54 “bitstrength” password
- ~1.5% of Yahoo users forget their passwords each month (!)

Internet users ditch “password” as password, upgrade to “123456”

Contest for most commonly used terrible password has a new champion.

by **Jon Brodkin** - Jan 20 2014, 4:00pm GMT

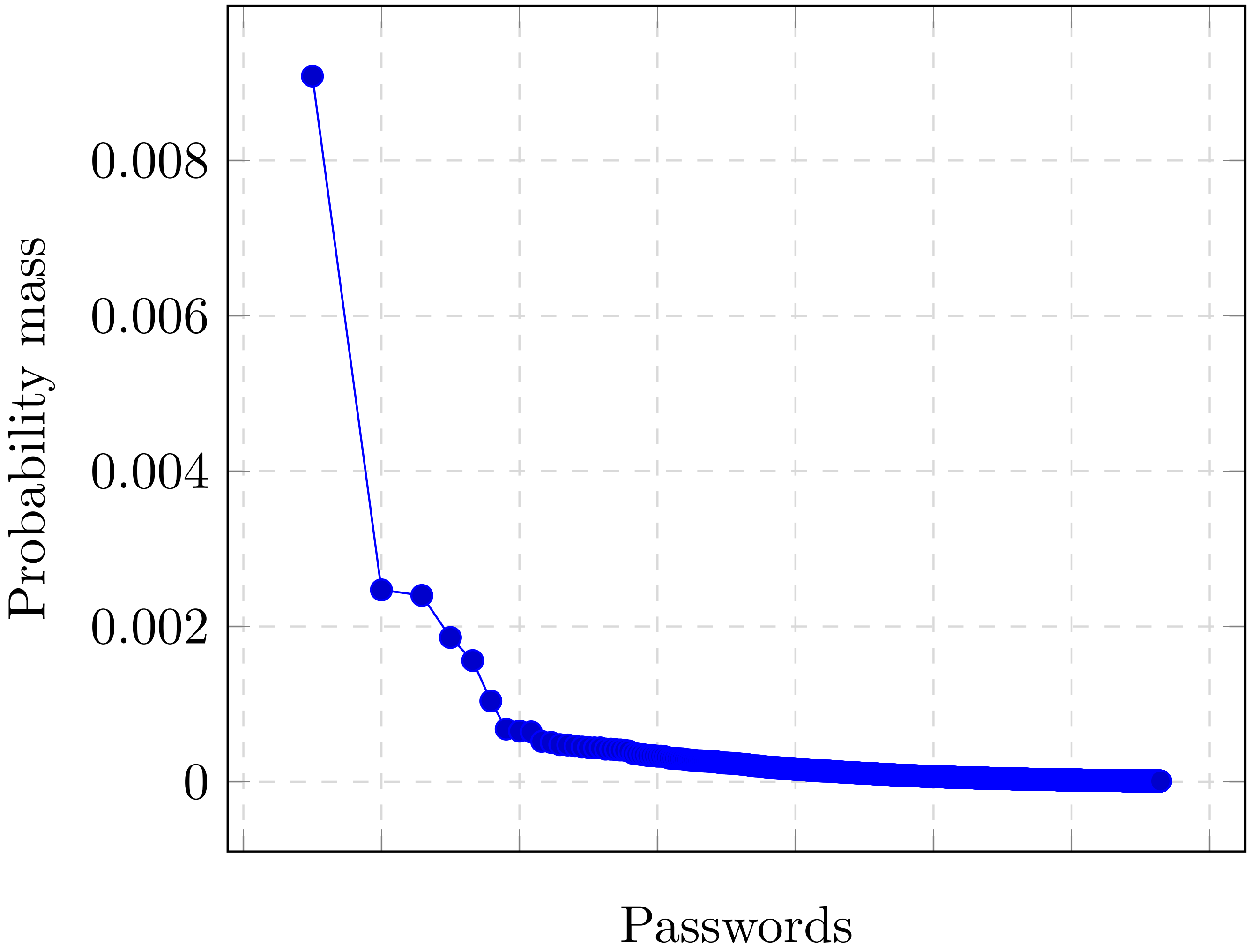
290729	123456
79076	12345
76789	123456789
59462	password
49952	iloveyou
33291	princess
21725	1234567
20901	rockyou
20553	12345678
16648	abc123
16227	nicole
15308	daniel
15163	babygirl
14726	monkey
14331	lovely

Rockyou data breach:
32 million social gaming accounts

Most common password used by almost 1%

[Bonneau 2012]
69 million Yahoo! Passwords
1.1% of users pick same password

Rockyou empirical probability mass function



(Only first 5,000 points shown)

Bonneau Yahoo password study

- Instrument login infrastructure
  - 69 million accounts monitored
- Hash passwords with key H(K,pw) and store result in histogram
- Throw away K
  - Can’t do brute-force attacks later on
  - Only learn empirical distribution of passwords
- Also stored some demographic information
- How do we measure strength of password distribution?

Password strength metrics

- Florencio and Herley approach?
  - Alphasize(pw) = sum of the sizes of character classes observed in password
    - Hello12! Has alphabet size = 26 + 26 + 10 + 22 = 84
  - Bitstrength(pw) = Alphasize(pw)$^{\text{len(pw)}}$
- Simpler than classical NIST entropy estimate

Password strength metrics

Let $\mathcal{X}$ be password distribution.
Passwords are drawn iid from $\mathcal{X}$
N is size of support of $\mathcal{X}$
p_1 , p_2 , …, p_N are probabilities of passwords in decreasing order

Shannon entropy:

$$H_1(\mathcal{X}) = \sum_{i=1}^{N} -p_i \log p_i$$

Shannon entropy is poor measure (for password unpredictability)

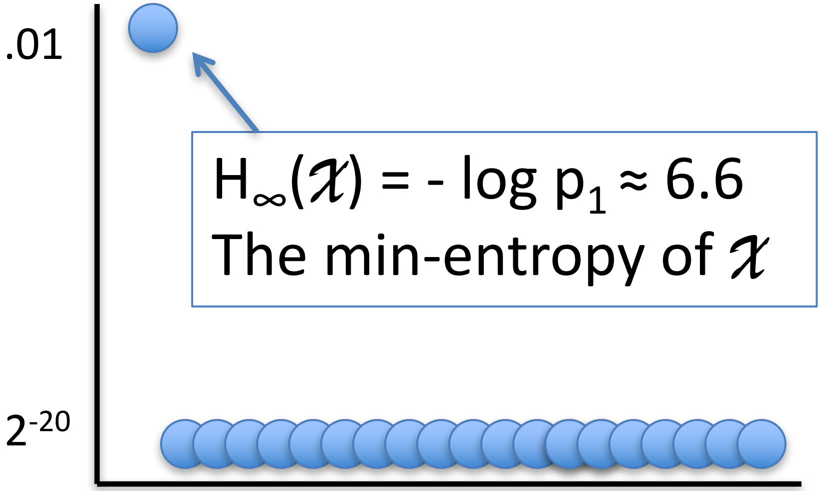
$N = 1{,}000{,}000$

$p_1 = 1 / 100$

$p_2 = (1 – 1/100)/999{,}999 \approx 1 / 2^{20}$

…

$p_N = (1 – 1/100)/999{,}999 \approx 1 / 2^{20}$



$H_\infty(\mathcal{X})$ = - log $p_1 \approx 6.6$
The min-entropy of $\mathcal{X}$

$H_1(\mathcal{X}) \approx 19$

19 bits of “unpredictability”. Probability of success about $1/2^{19}$

What is probability of success if attacker makes one guess?

Shannon entropy is almost never useful measure for security

Password strength metrics

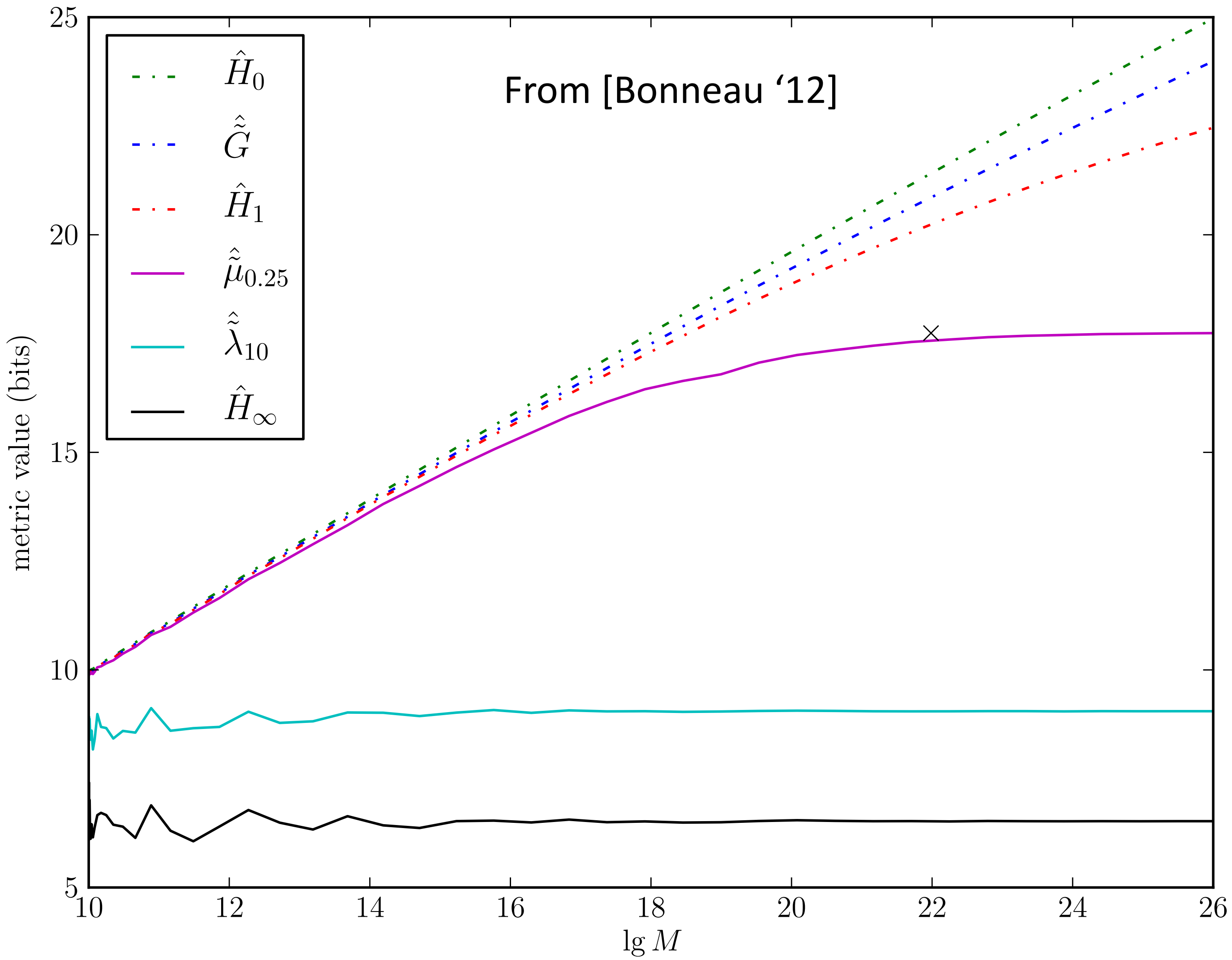
Beta-success rate:

$$\lambda_\beta(\mathcal{X}) = \sum_{i=1}^{\beta} p_i \qquad \tilde{\lambda}(\mathcal{X}) = \log(\beta/\lambda_\beta(\mathcal{X}))$$

Alpha-work-factor:

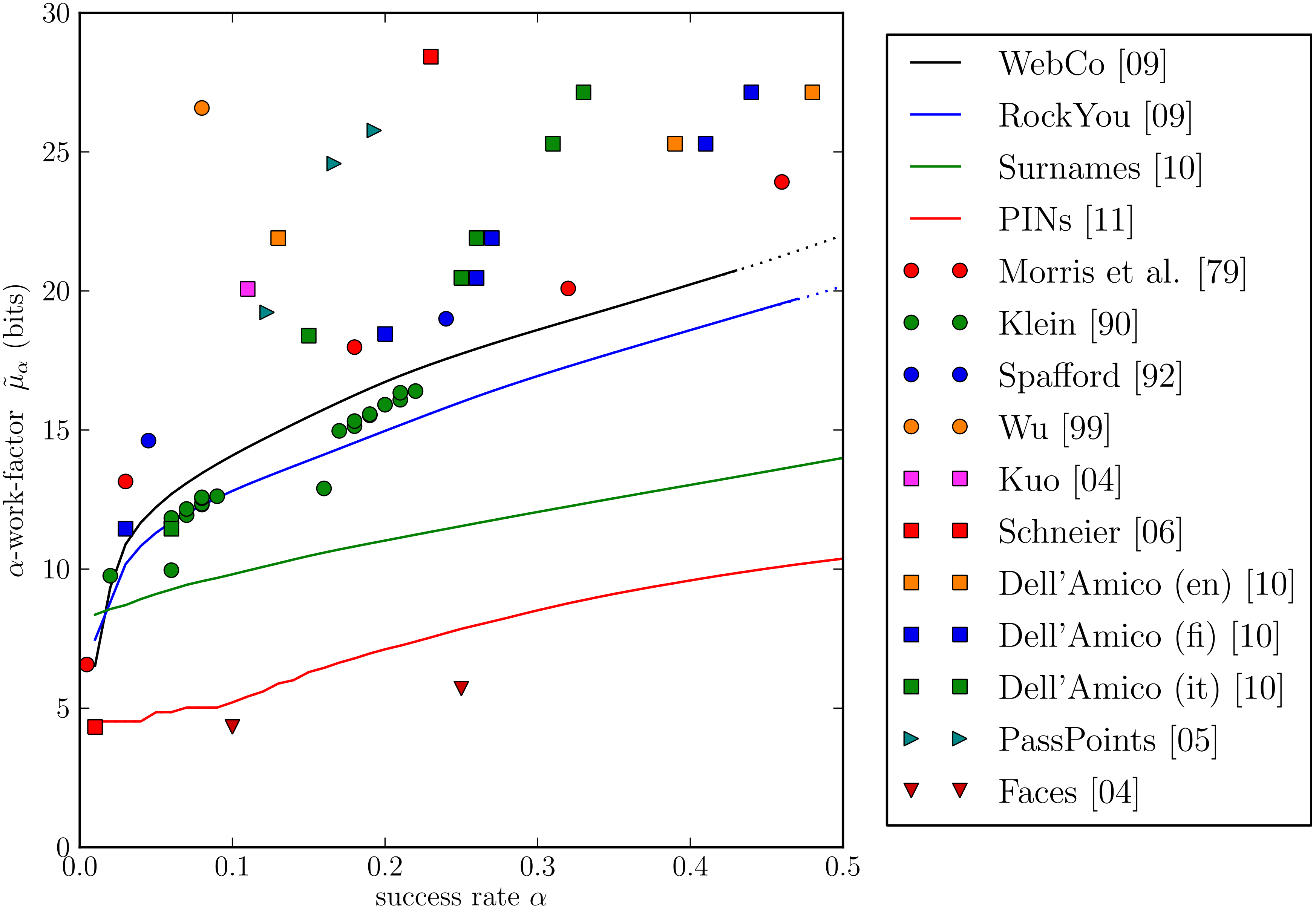
$$\mu_\alpha(\mathcal{X}) = \min\left\{ j \;\middle|\; \sum_{i=1}^{j} p_i \geq \alpha \right\}$$

$$\tilde{\mu}_\alpha(\mathcal{X}) = \log(\mu_\alpha(\mathcal{X})/\lambda_{mu_\alpha}(\mathcal{X}))$$



From [Bonneau '12]

Figure 3. Changing estimates of guessing metrics with increasing sample size M. Estimates for H_∞ and $\tilde{\lambda}_{10}$ converge very quickly; estimates for $\tilde{\mu}_{0.25}$ converge around $M = 2^{22}$ (marked ×) as predicted in Section V-A. Estimates for H_0, H_1, and $\tilde{G}$ are not close to converging.



From [Bonneau '12]

Bonneau takeaways

- Use appropriate strength measures for password distributions
- Yahoo study: people pick lousy passwords
- What does Bonneau paper not give us?

Brute-force attacks

- **Offline brute-force attacks**
  - Compromise database
  - E.g.: “cracking” via dictionary attacks
  - *Countermeasures*: hash passwords with purposefully slow-to-compute cryptographic hash function

    (was: MD5, SHA-1   now: argon2, scrypt)
- **Online brute-force attacks**
  - E.g: Submit guesses to web site
  - *Countermeasures*: Rate limit, account lockout

Building good password crackers

Brute-force guessers
- try all strings of a certain length

Dictionary guessers
- Try only common words

JohnTheRipper:
Dictionaries of common words +
mangling rules

Eg: add digit to end: pw -> pw1

Also has brute-force mode

A better guesser would:
Output list of passwords in order of likelihood

Understanding password strength

(1) Develop probabilistic model of passwords

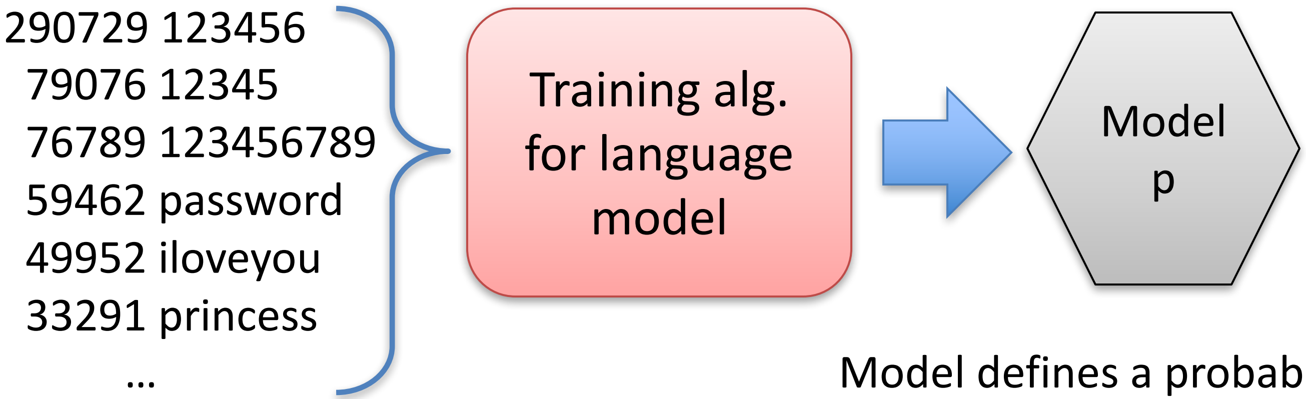
pw_1, pw_2 , … , pw_N

$p(pw_i) = p_i$ = probability user selects password pw_i

$$\sum_i p_i = 1$$

(2) Use p to educate brute-force crackers, strength meters, user interfaces

Train models from leaked passwords



290729	123456
79076	12345
76789	123456789
59462	password
49952	iloveyou
33291	princess
…	

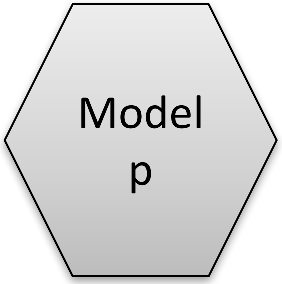
Training alg. for language model

Model p

Model defines a probability distribution over passwords. Can use to:

- sample passwords according to distribution
- enumerate passwords in order of likelihood

Trivial model is just the empirical CDF of the histogram itself

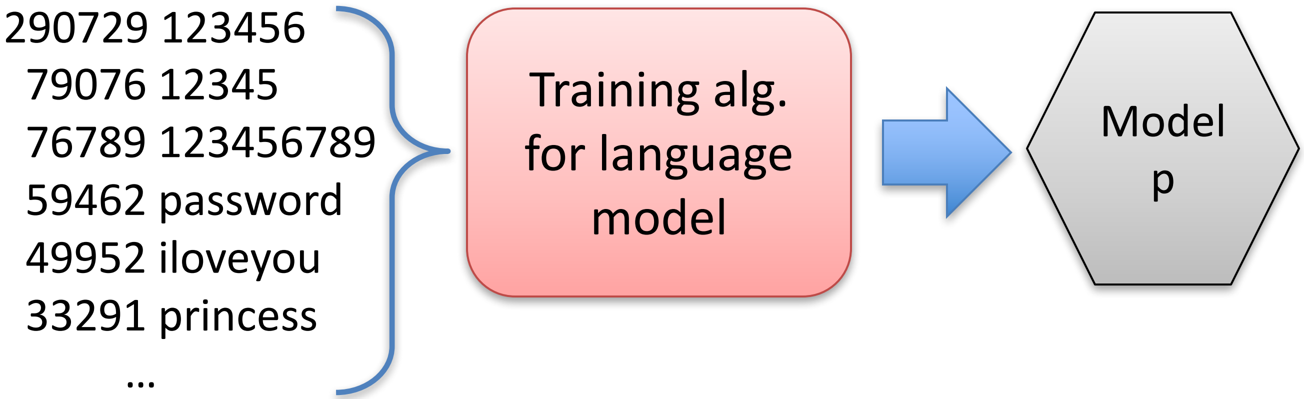


Model p

290729	123456
79076	12345
76789	123456789
59462	password
49952	iloveyou
33291	princess
…	

Supports all the above
Generalizability is quite poor
ML people would say this model is overfit

Train models from leaked passwords



290729	123456
79076	12345
76789	123456789
59462	password
49952	iloveyou
33291	princess
…	

Probabilistic context-free grammar (PCFG)

[Weir et al. “Password Cracking Using Probabilistic Context-free Grammars” 2009]

CFG with probability distribution associated to each rule

Fix a CFG, then learn probabilities by training on passwords

We can encode a string by its parse tree, the
tree represented by probabilities in PCFG CDF

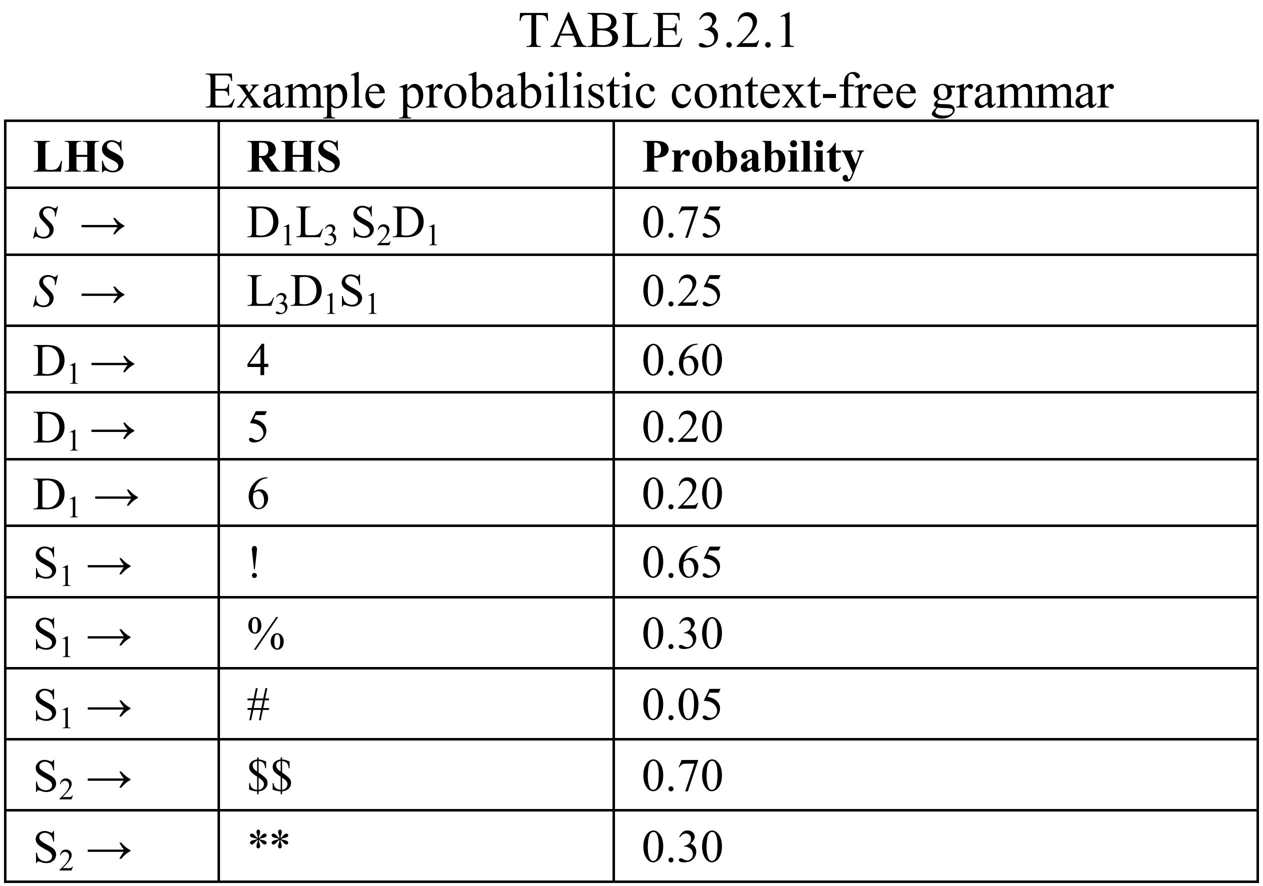


TABLE 3.2.1
Example probabilistic context-free grammar

LHS	**RHS**	**Probability**
$S \rightarrow$	$D_1L_3\ S_2D_1$	0.75
$S \rightarrow$	$L_3D_1S_1$	0.25
$D_1 \rightarrow$	4	0.60
$D_1 \rightarrow$	5	0.20
$D_1 \rightarrow$	6	0.20
$S_1 \rightarrow$	!	0.65
$S_1 \rightarrow$	%	0.30
$S_1 \rightarrow$	#	0.05
$S_2 \rightarrow$	$$	0.70
$S_2 \rightarrow$	**	0.30

S -> $L_3D_1S_1$ -> L_34S_1 -> $L_34!$

Pr[$L_34!$] = 0.25 * 0.60 * 0.65 = 0.0975

With good training data: Works better than JtR

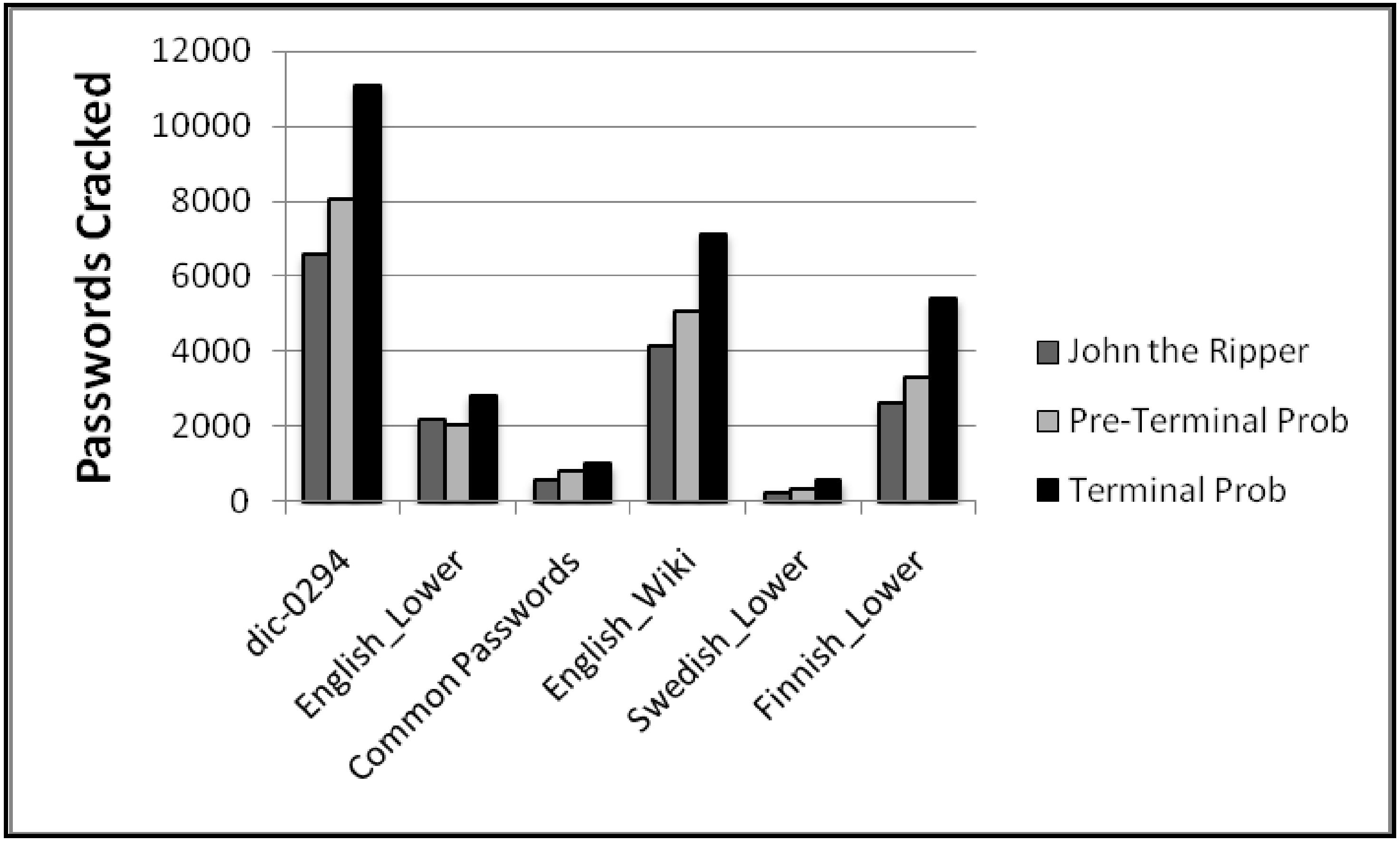
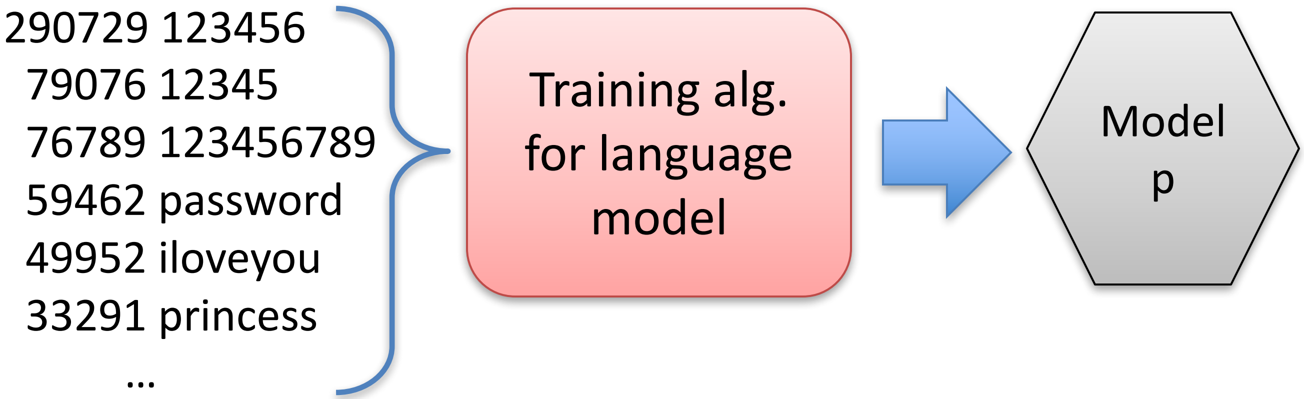


Fig. 4.4.1. Number of Passwords Cracked. Trained on the MySpace Training List. Tested on the MySpace Test List

Train models from leaked passwords



290729 123456
79076 12345
76789 123456789
59462 password
49952 iloveyou
33291 princess
…

Training alg. for language model

Model p

Probabilistic context-free grammar only one NLP modeling approach

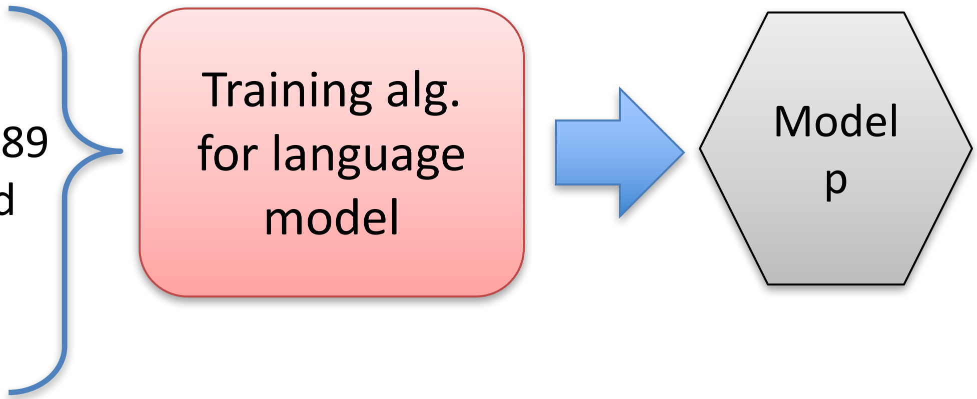
n-gram Markov models another popular choice.:

$$\Pr\left[\, w_1 w_2 \cdots w_k \,\right] \approx \prod_{i=1}^{k} \Pr\left[\, w_i \;\middle|\; w_{i-(n-1)} \cdots w_{i-1} \,\right]$$

[Ma et al. ’14] show carefully chosen Markov model beats Weir et al. PCFG

Train models from leaked passwords

290729	123456
79076	12345
76789	123456789
59462	password
49952	iloveyou
33291	princess
…	



Training alg. for language model

Model p

Neural network approach of [Melicher et al. 2016]

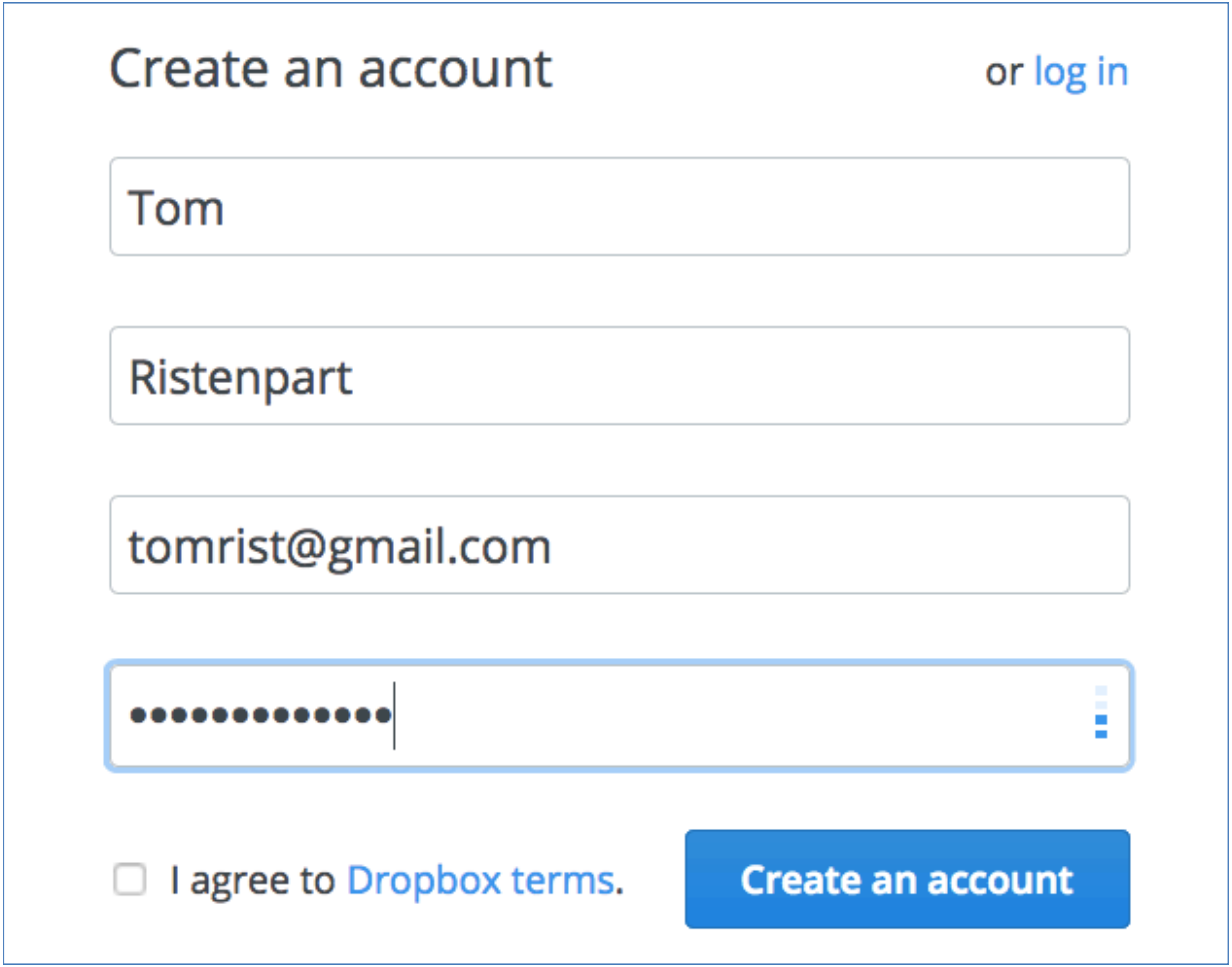
Use Long short-term (LSTM) recurrent neural network trained from large number of leaks (RockYou, Yahoo!, many others)

They primarily target using it as a strength meter:

For any pw, use $p(pw^*)$ to estimate the guess rank $|S(pw^*)|$

$$S(pw^*) = \{ pw \mid p(pw) > p(pw^*) \}$$

Can estimate using Monte-Carlo techniques [Dell'Amico, Filippone '15]



Create an account

or log in

Tom

Ristenpart

tomrist@gmail.com

●●●●●●●●●●●●●

□ I agree to Dropbox terms.

The research landscape since 1979…

- **Understanding user password selection**
  - Measuring password strength [see citations in Bonneau paper], [Li, Han `14], [CMU papers]
  - Measuring password reuse
- **Usability**
  - Strength meters, requirements, etc. [Komanduri et al. ‘11] [Dell’Amico, Filippone ‘15] [Wheeler ‘16] [Melicher et al. ‘16]
  - Password expiration [Zhang et al. ‘12]
  - Typo-tolerance [Chatterjee et al. `16]
- **Password transmission, login logic**
  - Single sign-on (SSO) technologies
  - Password-based authenticated key exchange [Bellovin, Merritt ‘92]
- **Password hashing**
  - New algorithms [PKCS standards], [Percival ’09], [Biryukov, Khovratovich ‘15]
  - Proofs [Wagner, Goldberg ‘00] [Bellare, Ristenpart, Tessaro ‘12]
- **Improving offline brute-force attacks**
  - Time-space trade-offs (rainbow tables) [Hellman ’80], [Oeschlin ‘03], [Narayanan, Shmatikov ‘05]
  - Better dictionaries [JohntheRipper], [Weir et al. ‘09], [Ma et al. ‘14]
- **Password managers**
  - Decoy-based [Bojinov et al. ’10], [Chatterjee et al. ‘15]
  - Breaking password managers [Li et al. ‘14] [Silver et al. ’15]
  - Stateless password managers [Ross et al. ’05]