Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms

Authors:
Patrick Gage Kelley, Michelle L. Mazurek, Richard Shay, Tim Vidas, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Julio López
Carnegie Mellon University, 2012

Presenter:
Savvas Kastanakis

Threat Model - Offline Attack

Attacker steals list of hashed passwords

Needs to guess passwords to crack them

Recent examples of such data breaches:

Gawker	1,300,000
Sony	25,000,000
Battlefield Heroes	550,000
Sega	1,300,000

This is considered a serious vulnerability, since many users reuse passwords, allowing "Malories" to access accounts in many platforms

Guessing Strategy of Attacker

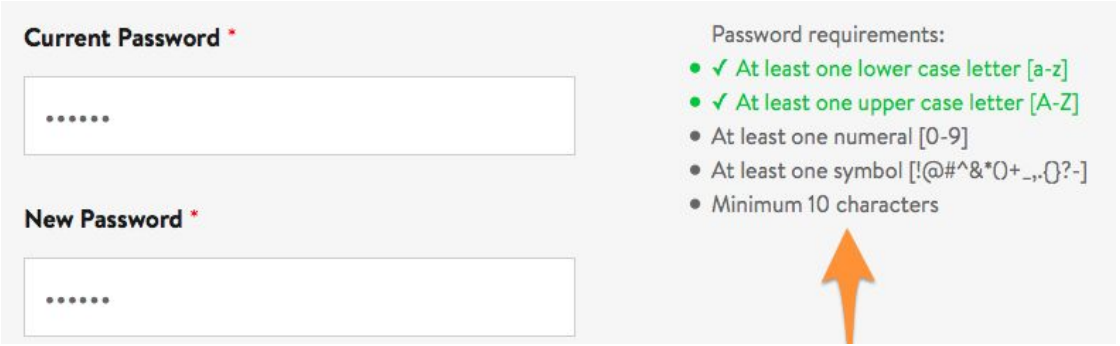
- Dumb attacker
  - aaaaaaaa
  - aaaaaaab
  - aaaaaaac
  - ....
  - aaaaaaaz
- Smart attacker
  - 123456789
  - password
  - iloveyou
  - princess
  - 87654321

- **Smart attacker uses data to crack passwords more quickly**

Composition Policies

Text-based passwords remain the dominant authentication method in computer systems

In response to this threat, we used composition policies, to make passwords harder to guess



Bad News: Composition Policies have grown increasingly complex

Contributions of this work

- Measured **guessability** across seven password composition policies
  - Threat model: offline attack
- Studied the impact of tuning and **data selection** on policy evaluation
  - What test data to use when evaluating password strength?
- Compare security metrics across policies
  - Correlate security with **usability**

How do we Quantify Effectiveness of Policies?

1. Entropy (based on information theory)
    a. Password entropy is a measure of the strength of a password based on information theory.
        i. Represents the maximum number of guesses a brute-force method would require to guess a given password
        ii. password : 18 bits / sapsword : 24 bits / Sapsword! : 30 bits
    b. Doesn't rely on empirical data on user behavior
2. Guessability (based on empirical analysis)
    a. Use password guessing tools to characterize the time needed to crack a password
    b. **Lack of available password sets**, hidden by organizations/enterprises

Threat model in this project - Offline Attack

- Offline attacker that can make up to 50 trillion guesses

- Attacker learns from training data
  - Leaked data plus collected passwords

- Attacker has limited knowledge of the target policy

Study Design

- Imagine that your main email service provider has been attacked, and your account became compromised
- You need to create a new password for your email account, since your old password may be known by the attackers (**guessability**)
- We will ask you to use this password in a few days to log in again so it is important that you remember your new password.
- Please behave as you would if this were your real password!
- Return in two days and insert the password (**usability**)

Guessability

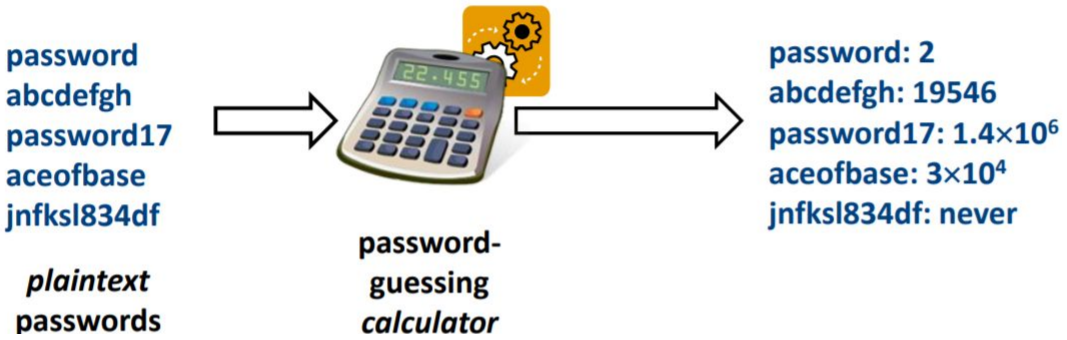
New measure of password strength: **Guess Number**

Bob's Password	**Attackers Guesses**	**Guess Number**
iloveyou123	1.password	3
	2.sapsword	
	3.iloveyou123	
	4.helloworld	

Guess-number Calculators

A calculator function maps a given password to the number of guesses required to guess that password

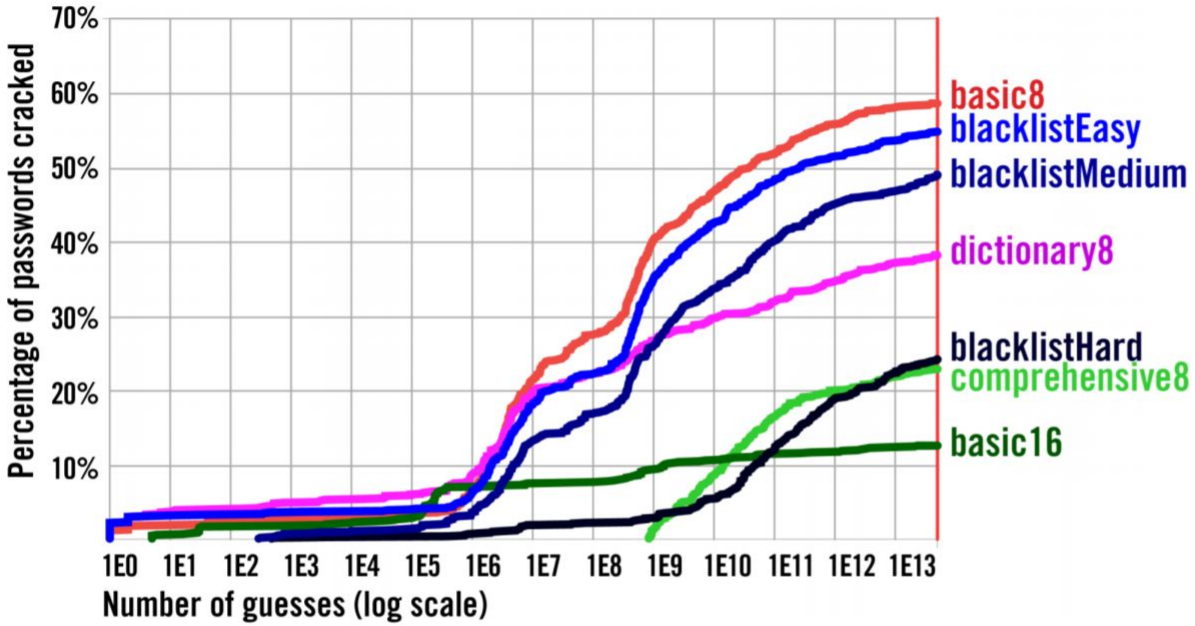
The output is the guess number of the password



Policies

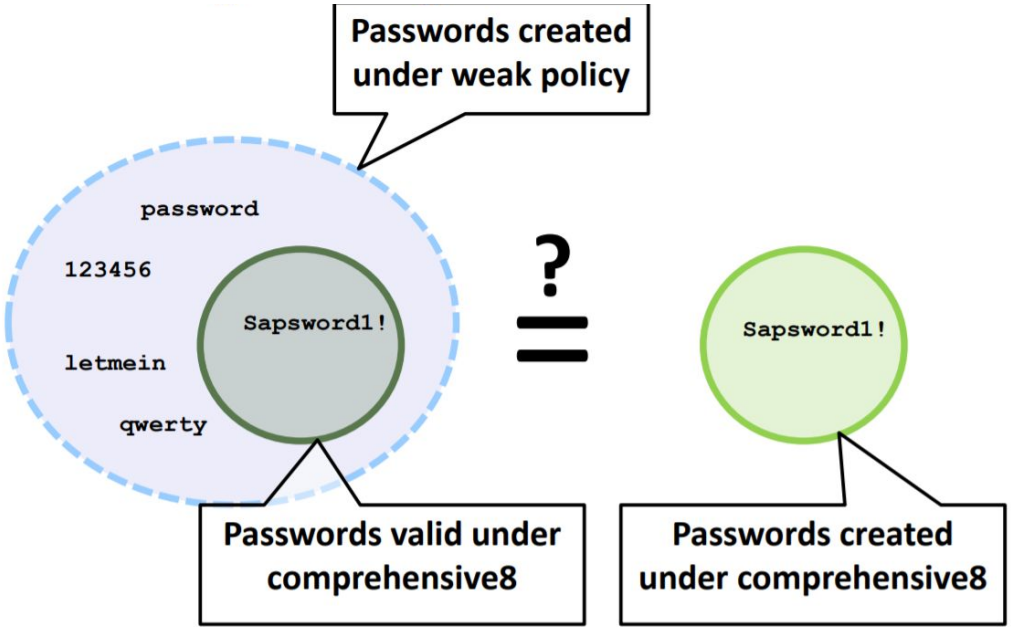
- **basic8**: "Password must have at least 8 characters."
- **basic16**: "Password must have at least 16 characters."
- **dictionary8**: "Password must have at least 8 characters. It may not contain a dictionary word." -- Free Openwall list dictionary.
- **comprehensive8**: "Password must have at least 8 characters including an uppercase and lowercase letter, a symbol, and a digit. It may not contain a dictionary word." -- Free Openwall list dictionary
- **blacklistEasy**: "Password must have at least 8 characters. It may not contain a dictionary word" -- from the Unix dictionary
- **blacklistMedium**: Same as the blacklistEasy condition, except use of the paid Openwall list.
- **blacklistHard**: Same as the blacklistEasy condition, except we used a 5B word dictionary created using the Weir algorithm

Guessability across 7 policies

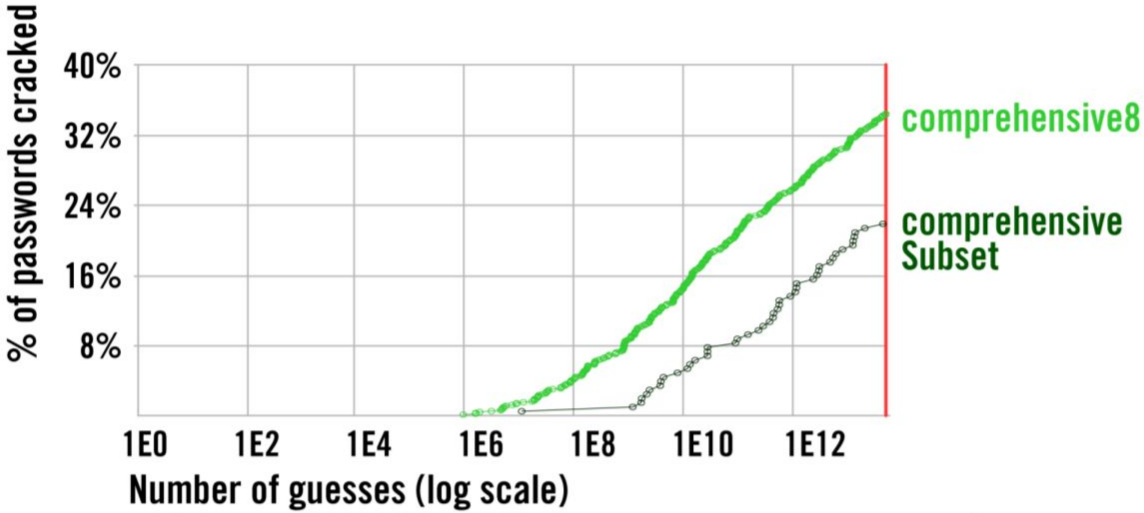


Choosing the right test data

- Providing random test data on a model, will return erroneous results
- We have to pick passwords that comply with the target password policy, such that we have accurate predictions



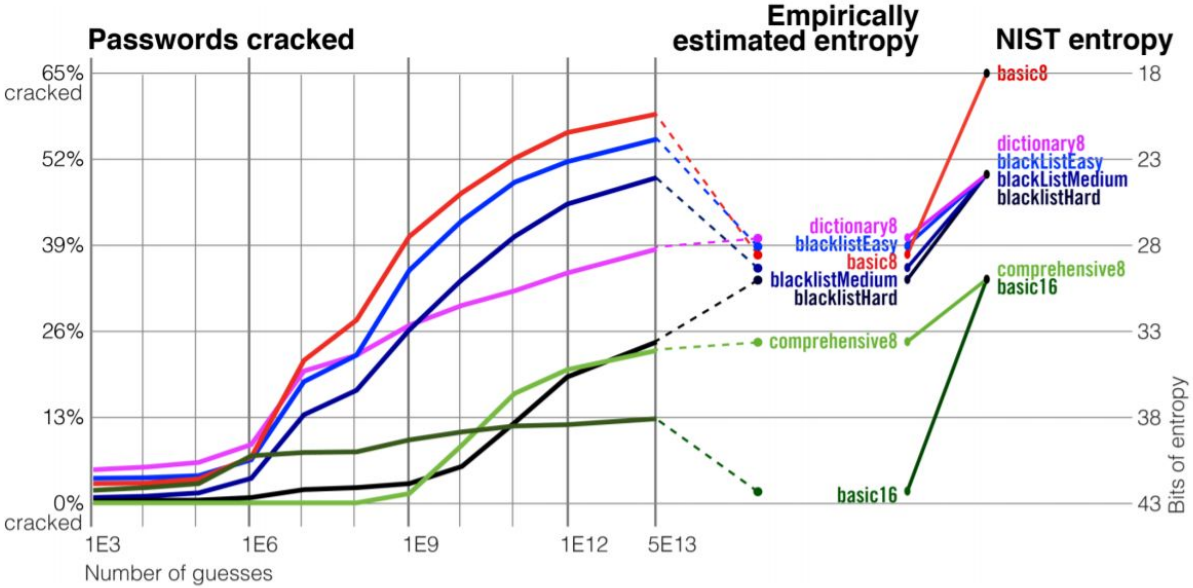
Selecting correct test sets is important



Carefully choosing test passwords is critical when evaluating policies

Guessability VS Entropy

- Although both measures of **entropy** provide a rough ordering among policies, they **do not always correctly classify guessability** (see for example dictionary8)
- They **do not effectively measure how much additional guess resistance one policy provides as compared to another**, since policies are clustered in one point (in contrast with password guessability)



Usability - Basic16 vs Comprehensive8

★ Basic16 is more usable

★ Fewer participants wrote down password (50% vs. 33%)

★ Self-reported difficulty and annoyance was lower

Basic16 appears to be more secure and more usable than comprehensive8

Take away message

- Picking a large but memorable password can trouble attackers more than any other policy
- Complex policies are tricky to analyze:
  - Need high-quality training data (usually these data are not revealed by System Administrators)
  - Important to choose test data carefully
- Password entropy provides only a rough correlation with guess resistance and is unable to correctly predict quantitative differences in guessability among password set