CPSC 530: Information Theory and Security
Fall 2017

Three Questions

Estimating Password Strength

Group 3

Niroojen Thambimuthu	10153928	BSc in Computer Science
Mark De Castro	10109634	BSc in Computer Science
Minh Tran	30017773	BSc in Computer Science
Masih Sadat	10066329	BSc in Computer Science

Three Questions

(1) What are some common flawed policies/guidelines that websites use when suggesting passwords to their users? List three such policies and briefly explain why they are “flawed.”

Answers can be any three among the following:

- **must have a minimum length** – an attacker will know the minimum length of a user’s password and can start guessing at that specified length
- **must add special characters/symbols** – an attacker will include symbols when guessing passwords; also, people tend to use ‘l33t’ (leet) when using symbols
- **must add uppercase/lowercase characters** – an attacker will know that a password is a combination of uppercase and lowercase letters; also, people tend to add uppercase letters at either the beginning or end of a password
- **must add a number/digit** – an attacker will know that a password will contain digits; also, people tend to add digits at either the beginning or end of a password

(2) What is the *zxcvbn* tool and its advantages over common password meters? List at least three of these advantages.

- The *zxcvbn* tool is an alternative password strength estimator that uses leaked passwords/dictionaries to simulate guessing attacks in order to measure a password’s strength. Its advantages over other password meters are (*answers can be any three among the following list below; we will present this whole list in our presentation*):
  - requires minimal storage space in order to run
  - can be adopted with 4 lines of code / easy to adopt
  - runs in milliseconds / runs fast
  - can be downloaded in seconds
  - works as-is on web browsers, iOS and Android

(3) What are the three phases that the *zxcvbn* tool goes through when measuring a password’s strength? Briefly describe each.

- **match**: given an input, find all possible patterns that match the given password
- **estimate**: estimate/calculate the strength/entropy of each of the matched patterns
- **search**: given all the matched patterns, find the pattern that is the simplest/has the lowest entropy