Reimagining Password Creation: Teaching Strength Through Prediction

Cherise McMahon

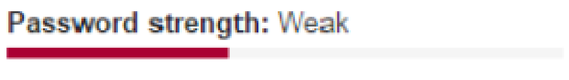
ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Problem

Whether from a lack of knowledge, laziness, misconceptions, or other issues, users struggle to create unpredictable passwords.

P a s s w o r d 1 2 3 !

To counteract this issue of prediction, the objective of this research is to learn **if adding negative feedback during password creation helps users create stronger passwords**.



This study will also focus on **if users learn better password practices from interacting with a tool.**

Methodology

By creating a tool with **predictive completion** and **negative feedback**, users will **create strong passwords and learn better strength habits.**

Feedback is given through warnings and non-suggesting complexity requirements. The tool was kept simple and straight forward.

To avoid giving positive feedback for the complexity requirements, they instead fade out once criteria is met rather than show a green check mark.

✓ Number	Number
× Special Character	Special Character
✓ Upper Case Letter	Upper Case Letter
× At Least 8 Characters	At Least 8 Characters

Warnings were displayed as needed in a yellow box.

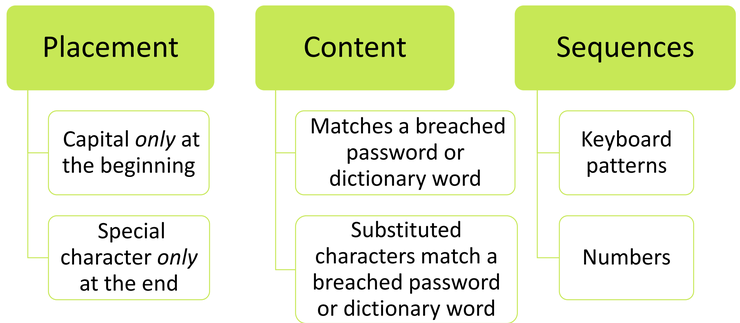
WARNINGS:

Are **limited** to three
To avoid overwhelming
The user **…**

Warnings

In order to best inform the warning feedback that users got, a survey was performed to see if specific or general warnings were best. It turned out that **language specificity didn't matter much, so simple language warnings were used.**

Things that trigger warnings are:



Prediction

The **invisible dictionary** is the basis of the prediction algorithm. This dictionary is made up of the 20 thousand most common English words and 14 million breached passwords.

In order to have any **substring** have the potential to be **autocompleted**, the dictionary was loaded into redis as substrings.

for all i = 0…length(word):
map word.substring(0, i) ⟶ word

This method creates a lot of entries, but it means **every lookup is immediate for any substring** that exists in the invisible dictionary.

If a substring exists already, it is not overwritten. This means the database needs to be loaded based on priority, which it is why **it is important to have the text files be ordered by high frequency.**

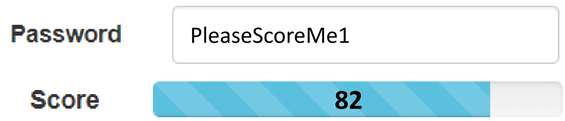
After a user types **four** characters, the prediction starts running.

s p i d ⟶ ● ● ● ● e r m a n

Testing

Participants were asked to create **two passwords**, one with the tool and one without.

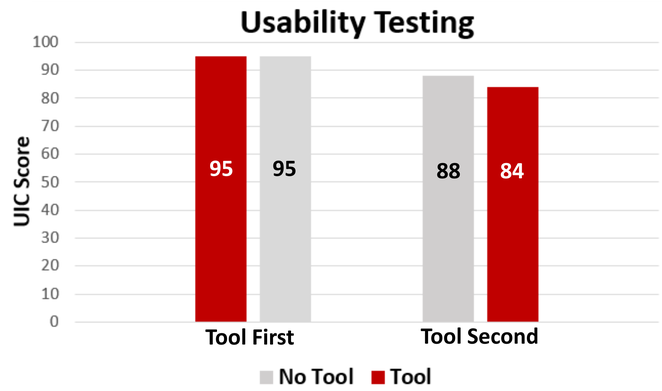
They were randomly assigned on the order they used the tool. **This allowed potential to see how the tool both helped users and influenced their password creation methods.**



Passwords were scored with the University of Illinois at Chicago's password strength meter for a numerical comparator.

Finally, participants answered a post-test questionnaire about their experience with the tool.

Results



Unexpectedly, it seems that users who use the tool first create stronger passwords than those who use it second.

Participants who used the tool second created passwords that were, on average, a **character shorter than the ones they created first.** Participants who used the tool first **had the shortest average passwords**, but they **maintained the high complexity.**