Refining Graphical Password Strength Meters for Android Phones

Susanna Heidt
United States Naval Academy
m172610@usna.edu

Adam J. Aviv
United States Naval Academy
aviv@usna.edu

Acknowledgment:
This work was support in part by ONR grants N001416WX01664 and N0001416WX01494.

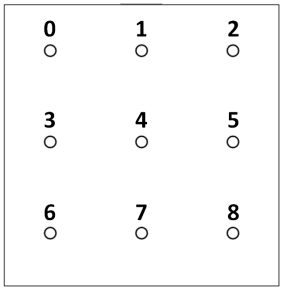
Abstract

The purpose of our research is to improve upon current strength meters by analyzing their computations. These strength meters incorrectly assume a linear relationship between individual features of a pattern, such as pattern length, and its strength according to each meter's strength metric. Instead we designed our own strength meter- tested via a web browser- that uses a pattern's guess number to determine its strength. Our meter visualizes for the user the strength of his/her graphical password by displaying the real time guess number as well as a gradient meter bar. We propose that this meter should be implemented in future tests because it matches the security metric of guessability, which is the current best known representation of pattern strength.

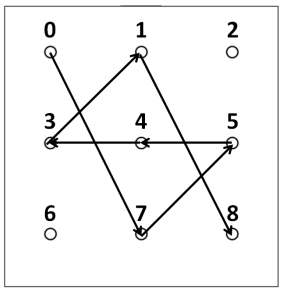
Introduction

The use of graphical passwords is a relatively new form of phone authentication and as such researchers have been exploring ways to make it stronger. In "Is Bigger Better?" by *Aviv et. al*, they looked into whether a larger 4x4 grid space encouraged users to create more secure passwords than the current 3x3 grid [2]. Dumnphy and Yan studied the effect on complexity of using a picture under the actual pattern [3]. There has also been different implementations of a "strength meter" that visually shows the user how secure his/her password is, which is what we are most interested in. Past research has proven that the presence of some sort of visual meter does in fact help users to create more secure

Figure 1



(a)



Example of valid graphical password

(b)

passwords [4]. However, the purpose of our research was not to verify this fact, but rather we propose a new meter based on guessability. First, we looked at the factors that contributed to the other meters and implemented them via a web browser. Then we added our own guessability strength meter to have a side by side visual comparison test. Finally, we show that our meter is the best representation of pattern strength because it neither picks certain features arbitrarily to use in its computation nor does it assume a purely monotonic relationship between the feature and strength.

Background and Related Work

Today millions of cell phone
users take advantage of Android's graphical password feature to lock their phones. Graphical passwords have gained popularity over other methods of authentication, such as text based passwords and personal identification numbers (PINs), because the visual aspect makes them easier for the user to remember while at the same time maintaining complexity and thus security.

Earlier we used a brute force method
to determine that there are 389,112 total possible patterns that can be made from Android's pattern lock rules. As you can see in Figure 2a, the user starts with a blank 3x3 grid of points, which we have numbered from 0 to 8 starting in the upper left in order to keep track of what points have been used for a particular pattern. While drawing the pattern, one must make sure to contain at least four points such that no point is repeated. Additionally, all points along the path must be included. For example, in order to move from point 8 to point 0, point 4 must be touched and therefore part of the pattern. Figure 2b shows an example of a valid graphical password a user may choose.

Strength Meters

In order to design and implement our own meter, we first analyzed how previous research created strength meters. After reviewing Song's[4], Sun's[5], and Andriotis's[1] respective papers, we implemented each one so that we could see how they compared with each other and eventually our own meter. However, in order to understand how each meter's score was computed, one must first be able to identify the features that were used.

Meter Pattern Features and Implementations
Every pattern can be
classified into different groups based on its features, which are described below. It is important to note that not every feature has to be present for a pattern to be valid and that not every feature is taken into consideration to compute a certain meter.

- Start (S_p) - start point of pattern *p*
- Length (L_p) - number of points touched in pattern *p*. For example, pattern 6.7.3.8.5 has length $L_p = 5$.
- Crosses (C_p) -
  number of crosses in pattern *p*. A cross occurs when two non-consecutive line segments have a common point.
- Non-adjacent
  (A_p) - number of non-adjacent segments in pattern *p*. Non-adjacent segments exist when one covers another.
- Knight moves (K_p) - number
  of "k-moves" in pattern *p*. Much like the knight moves on a chessboard, it involves moving one point horizontally and two vertically (or vice versa) to reach the next point.
- Turns (T_p) - number
  of turns in pattern *p*. Think of it as a direction change.
- Euclidean
  distance (E_p) - sum of Euclidean distances between points. Any horizontal or vertical move is one unit apart, so for every segment apply the Pythagorean Theorem.

Figure 2

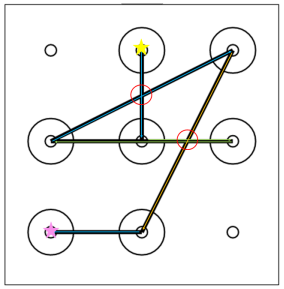
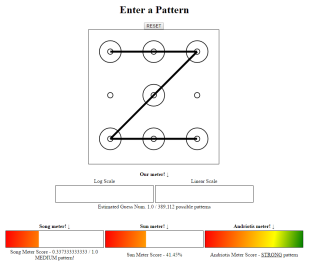
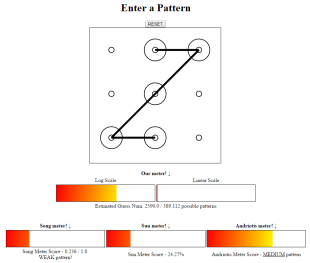


Figure 3



(a)



(b)

- Maximum vertical or horizontal distance (D_p) - sum of segment's maximum vertical or horizontal distance.
- Non-repeated segments (N_p) - ratio of non-repeated to total segments.

Examples of all of these features are shown in Figure 2.

Song Meter *Song et. al* developed a meter, which we refer to as the Song Meter, based on three features- D_p, N_p, and C_p - to produce a "Song Score" from 0 to 1. The Song Score is computed using the following equation

$$M_p = 0.81(\frac{D_p}{15}) + 0.4N_p + 0.15(\frac{min(C_p,5)}{5})$$

The pattern is then classified as being either "weak," "medium," or "strong" if the score falls within the ranges 0.00-0.33, 0.34-0.67, and 0.68-1.00 respectively[4].

Sun Meter Similar to the Song Meter, the Sun Meter by *Sun et. al* also uses the number of crosses, but analyzes the pattern's length, Euclidean distance between points, and number of non-adjacent segments to determine the score.

$$PS_p = L_p * \log_2(E_p + C_p + A_p)$$

Raw Sun Scores range from 6.340 to 46.807 so that the stronger the pattern is the greater the raw score. In order to put the Sun Score in context for the user, we display the score as a percentage so that a higher percentage score indicates a more secure pattern[5].

Andriotis Meter The Andriotis Meter by *Andriotis et. al* produces a maximum Andriotis Score of 18. The following equation computes a pattern's Andriotis score.

$$\Delta = N \cdot X = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} S_p \\ L_p \\ T_p \\ K_p \\ A_p \end{bmatrix}$$

Much like the Song Meter, the Andriotis Meter also uses the resulting score to determine whether the pattern is weak, medium, or strong. However, a weak pattern falls between 0 and 1 inclusive, a medium pattern's score is exactly 2, and any score greater than or equal to a 3 indicates a strong pattern[1].

Our meter Rather than base our meter's score purely on the pattern's features, we use a guessability metric instead to measure the pattern's strength by considering how many guesses it takes to guess the given pattern. For every pattern, we used the pattern's points in tri-grams to generate the Markov guess probability (as described and based on data previously collected in [2]). All of the patterns were then ranked from greatest to least probability so that the most likely guessed patterns have the highest probability and therefore a lower guess number. For example, pattern 0.1.2.4.6.7.8, shown in Figure 3, is the most common pattern. It has the highest Markov probability of 1.141, so its guess number is 1.

Browser Meter Implementation
We implemented all of the meters in a web browser so that as the user draws a pattern, each meter's score is displayed in real time and fills the gradient meter bar accordingly. For every meter, a database was created containing all possible patterns and their associated score in order to make the score look up and page loading faster.

For our guessability meter in particular, we created two gradient meter bars using different scales to visually show the strength of

a user's pattern. Since all of the other meters use a linear scale, we started with a linear one for our meter as well. However, we also created a meter using a base two logarithmic scale in order to better differentiate between the guessing numbers and their relative strengths. For example, even though there's a big difference between 40% and 90%, which would clearly show up visually on the linear scale, both should be considered secure and the logarithmic scale is able to convey that.

Monotonic Comparisons

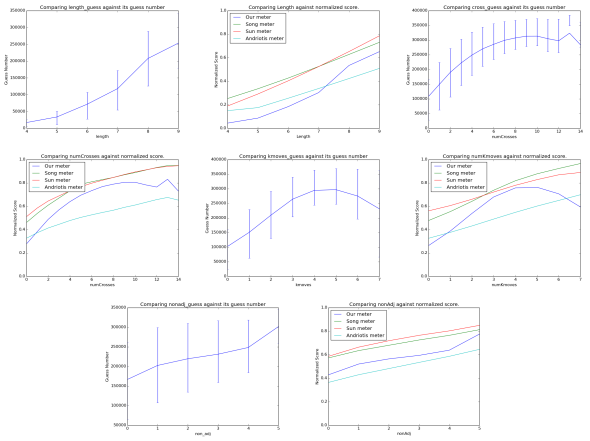
Based on the scores for every pattern from every meter, we first determined the relationship between each feature and the average guess number as seen in Figure 5. We also plotted the first and third quartile for every data point in order to better understand the spread. As we suspected, the features in relation to the guessability score is *not* strictly monotonic. The k-moves is one of the best examples of this because it shows that as the number of k-moves increase, the guessability and therefore the pattern strength does not necessarily increase. In fact, after 4 k-moves, pattern strength decreases.

We then plotted the relationship between the features and the normalized score of each meter in order to show exactly how different the other meters are in comparison to ours. In order to normalize each meter's score we converted set the maximum normalized score for any metric to one and adjusted accordingly. For example, the Sun Meter is originally out of 46.807 points, so to normalize, every pattern score is divided by 46.807. By plotting the meters on the same graph, as shown in Figure 5, one can see the monotonicity of the other meters in comparison to our own guess meter.

Conclusion and Future Work

Based on our monotonic comparisons, we assert that our meter matches the security metric of guessability, which is the best representation of pattern strength. Unlike the other meters,

Figure 5



we do not assume that a feature has a linear relationship with strength. For example, it is incorrect to say that just because pattern A has more k-moves than pattern B, pattern A is stronger.

We would like to further examine the differences between the meter scores first through more graphs and then with a user study. As for the graphs, we are interested in seeing how the normalized scores compare to each other for every pattern in previously determined sets of patterns, such as the self-report set, the pen-and-paper set, and the set of all 389,112 possible patterns.

We have also been thinking of designing a user study that analyzes the effect of a meter's existence. There have already been similar studies done, but what will make ours unique is the fact that the control group will look at a naive meter rather than nothing at all while inputting their graphical password. From there, we will also be able to determine if adding extra elements to the visual meter without any justification has an effect on the user.

REFERENCES

1. Panagiotis Andriotis, Theo Tryfonas, and George Oikonomou. Complexity metrics and user strength perceptions of the pattern-lock graphical authentication method. In *Human Aspects of Information Security, Privacy, and Trust*, pages 115–126. Springer, 2014.
2. Adam J Aviv, Devon Budzitowski, and Ravi Kuber. Is bigger better? comparing user-generated passwords on 3x3 vs. 4x4 grid sizes for android's pattern unlock. In *Proceedings of the 31st Annual Computer Security Applications Conference*, pages 301–310, 2015.
3. Paul Dunphy and Jeff Yan. Do background images improve draw a secret graphical passwords? In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 36–47, 2007.
4. Youngbae Song, Geumhwan Cho, Seongyeol Oh, Hyoungshick Kim, and Jun Ho Huh. On the effectiveness of pattern lock strength meters: Measuring the strength of real world pattern locks. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 2343–2352, 2015.
5. Chen Sun, Yang Wang, and Jun Zheng. Dissecting pattern unlock: The effect of pattern strength meter on pattern selection. *Journal of Information Security and Applications*, 19(4):308–320, 2014.