Quantifying the Security of Graphical Passwords: The Case of Android Unlock Patterns

Sebastian Uellenbeck, Markus Dürmuth, Christopher Wolf, and Thorsten Holz
Horst Görtz Institute for IT-Security, Ruhr-University Bochum, Germany
{firstname.lastname}@rub.de

ABSTRACT

Graphical passwords were proposed as an alternative to overcome the inherent limitations of text-based passwords, inspired by research that shows that the graphical memory of humans is particularly well developed. A graphical password scheme that has been widely adopted is the *Android Unlock Pattern*, a special case of the Pass-Go scheme with grid size restricted to 3×3 points and restricted stroke count.

In this paper, we study the security of Android unlock patterns. By performing a large-scale user study, we measure actual user choices of patterns instead of theoretical considerations on password spaces. From this data we construct a model based on Markov chains that enables us to quantify the strength of Android unlock patterns. We found empirically that there is a high bias in the pattern selection process, e. g., the upper left corner and three-point long straight lines are very typical selection strategies. Consequently, the entropy of patterns is rather low, and our results indicate that the security offered by the scheme is less than the security of only three digit randomly-assigned PINs for guessing 20% of all passwords (i. e., we estimate a partial guessing entropy $G_{0.2}$ of 9.10 bit).

Based on these insights, we systematically improve the scheme by finding a small, but still effective change in the pattern layout that makes graphical user logins substantially more secure. By means of another user study, we show that some changes improve the security by more than doubling the space of actually used passwords (i. e., increasing the partial guessing entropy $G_{0.2}$ to 10.81 bit).

Categories and Subject Descriptors

K.6.5 [**Security and Protection**]: Authentication; D.4.6 [**Operating Systems**]: Security and Protection

General Terms

Security; Human Factors

CCS'13, November 4–8, 2013, Berlin, Germany.

ACM 978-1-4503-2477-9/13/11 . . . $15.00.
http://dx.doi.org/10.1145/2508859.2516700.

Keywords

Passwords; Mobile Security; Android

1. INTRODUCTION

Authenticating a user is one of the central tasks in computer security, and it forms the basis for several other tasks as well. Text-based passwords are a common approach for authentication, but it is well-known (see for example [7]) that users typically choose weak passwords, and have problems to recall strong ones. As an alternative to text-based passwords, *graphical passwords* [4] have been proposed. These schemes are motivated by psychology research results suggesting that the human brain is particularly well-suited to remember graphical information [33].

In practice, very few graphical password schemes have been adopted and presumably the most widely deployed one is used in Android systems. Main reason is that Android is the most popular OS for modern smartphones. In the so called *Android Unlock Pattern scheme*, the user is presented a 3×3 grid and the secret (password) of a user is a drawing on that grid (i. e., a sequence of lines connecting the dots). During enrollment, a user has to choose a pattern and during the authentication phase, he has to recall his pattern and draw it on the screen. Android's Unlock Pattern is a modified version of Pass-Go [35] with minor adoptions to accommodate for the size of typical mobile devices. In general, the usability of graphical password schemes is well understood [4]. However, security aspects of such schemes are hardly studied, with only a few exceptions (e. g., [18,20,23,36]). As such, the question remains how secure graphical passwords are *in practice* in comparison to a theoretical analysis of password spaces. The latter is usually the far more impressive number, but not that meaningful as a metric for the strength of the corresponding scheme.

In this paper, we conduct the first (to the best of our knowledge) large-scale user study on Android Unlock Patterns, with a total of 584 participants and approximately 2,900 patterns. By means of a pen-and-paper study, we interviewed 105 people to find out their strategy to chose patterns on an Android phone. Furthermore, we performed a study with 113 participants as part of a game in which people were asked to chose a pattern for a phone provided by us, and also to guess patterns used by other participants of the study. The collected data enables us to analyze how people pick patterns and to attack them in order to evaluate their strength. We performed an analysis based on Markov chains to quantify their security and found that Android Unlock Patterns offer less security than a three digit (deci-

mal) PIN. As our study only provides an *upper* limit on the entropy, the situation should be slightly worse in practice.

During our study, we found several typical strategies used by the participants for the pattern generation process (e. g., people often picked the top left corner as a starting point and prefer straight lines in their pattern). Drawing from this knowledge, we strive to improve the security of the scheme: by systematically changing the pattern layout in such a way that these typical strategies are thwarted, we encourage users to chose patterns in a more diverse way. However, we concentrate on small modifications that allow us to keep the overall design of the patterns to foster the roll-out of a potentially better graphical password scheme. Specifically, we experimented with different arrangements of the patterns, such as removing the top left point, arranging them in form of a circle, or a (seemingly) random pattern. We performed another user study with 366 participants to empirically verify which effect such changes in the pattern layout have in practice, and found that the entropy estimate for some patterns increases significantly. In one modified pattern, we removed the upper left corner as it is the source for most of the bias. However, it turned out that people simply started from the point to the right of it, so this modification did actually *worsen* the security and the entropy estimate decreased. We got a slightly better result by switching to a 4×3 grid and removing the left upper and right lower corner. We pursued a different approach with the two new patterns *Random* and *Circle*. In the first, a (seemingly) random pattern was presented that excluded straight lines between any 3-points, a clear top/left position, and also allowed easy transition from all 9 points to any of the remaining 8 points. In the second, we arranged the nine points in a circle instead of a grid. While the initial bias was in the same range as for Android Unlock Patterns, it turned out that the following *transitions* were more diverse: we found that people prefer the directly neighboring point, but also include a point slightly further away. In addition, the *Circle* pattern does not allow to stay "on one line" as opposed to a quadrangle. In particular, for the circle patterns this increased the overall entropy estimate.

In summary, we make the following contributions:

- We performed a user study on Android Unlock Patterns to find out how users typically chose patterns. While it is not surprising that Android Unlock Patterns fall short of reaching their theoretical strength, we are the first to actually quantify the strength of user chosen patterns on the Android platform.
- Based on the insights obtained during this study, we systematically change the pattern layout to remove bias in the pattern selection process to improve the security of the scheme. In another user study, we found that this simple re-ordering can greatly improve the entropy estimate of the patterns at basically no cost.
- Overall, we collected more than 2,800 patterns chosen by users. In total, more than 580 people participated in our study. On a broader scale, this work teaches us valuable lessons about user choice in selecting graphical passwords.

Paper Outline

The rest of this paper is structured as follows. In Section 2, we discuss related work, in particular on graphical password schemes. In Section 3 we consider the unmodified Android Unlock Patterns and provide details on the user study we used to collect a large number of patterns. We describe an attack based on Markov models in Section 4 and quantify the security given by those patterns. In Section 5, we identify several weak points of the original patterns, test a number of alternative (but similar) constructions, and quantify their security. We conclude this paper with a discussion of the results and an overview of future work in Section 6.

2. RELATED WORK

The security of (text-based) passwords has been studied as early as 1979 [27], and it has been realized early that passwords are susceptible to so called *dictionary attacks* as many users choose passwords from a small subset of all possible passwords that can be collected in a dictionary. Popular password guessers that perform dictionary attacks are *John the Ripper* [32] and *HashCat* [1], advanced password guessers are based on Markov models [28]. Several approaches have been tested to prevent users from choosing weak passwords. Best known are so-called *password rules*, which are deployed by most websites today, but have been shown to have limited effect [25]. More recent techniques use Markov models to measure password strength [13]. A good overview of passwords is given by Bonneau [7].

Numerous alternatives to text-based passwords have been proposed. One particularly interesting class is *graphical passwords*, which have the potential to offer better usability because the human brain is particularly well-suited to remember graphical information [33]. Graphical passwords are typically classified as *recall-based*, *recognition-based*, or *cued recall-based*, where Android Unlock Patterns fall into the first category. In the following we review some of the work on graphical passwords, and refer to the recent survey by Biddle, Chiasson, and van Oorschot [4] for more details.

Recall-based schemes.

The first recall-based graphical password scheme *Draw-A-Secret* (DAS) was proposed in 1999 by Jermyn et al. [24]. Users select one or more strokes on a 2D grid, and need to reproduce these strokes when logging in. BDAS [21] augments this scheme by adding a background image, which helps the user to choose more complex and in particular longer patterns. Robustness to small errors is improved with YAGP [22] and Passdoodles [38].

Most relevant for our work is the Pass-Go scheme [35], which uses the intersections of a 2D grid instead of the cells. To evaluate the strength and usability, the authors conducted an informal user study with 167 participants which showed that users chose passwords with an average length of approximately 17, which leads to an estimation for the size of the password space of 2^{109}. The participants attending to the study were students from two university courses. Therefore, the authors had chosen a very homogeneous set for their study. In contrast to this, we asked people from all faculties (engineering, humanities, medicine, and science) of the university to participate. As a result, we belief that our results better model reality. Furthermore, we asked more than two times as much people to participate in our study.

Variants of the Pass-Go scheme are PassShape [41], which use a different input interface and coding of the patterns, and a variant that additionally uses haptic input [29] in an attempt to counter shoulder-surfing attacks. GrIDsure employed patterns on a grid to implement one-time PINs

(see [6, 10]). The Android Unlock Patterns are a special case of the Pass-Go scheme with restricted grid size and restricted stroke count, see Section 3.1 for more details.

An attack specific to implementations on devices with touch screen are so-called smudge attacks [2], where residue from using the touchscreen reveals information about the pattern. While this was originally studied for the Android Unlock Patterns, similar problems should arise for most authentication schemes on touch-screens.

Recognition-based schemes.

The classical example for recognition-based schemes are PassFaces [30]. Users choose a set of (images of) faces, and need to select those among a number of decoy images for login. Their usability is well studied [11]. However, [18] found that the PassFaces scheme with user-chosen faces, which is the proposed standard, is quite insecure: user's choices of faces are very biased, basically towards the user's race and towards female faces.

The scheme Story [18] is similar to PassFaces, but users learn images of more general persons and objects. The scheme Deja Vu [19] uses images displaying "random art". A further interesting concept was proposed by Weinshall [39], where the user learns a (relatively large) set of images, and for login he solves a "graphical puzzle" based on these images, in an attempt to counter shoulder-surfing attacks. Unfortunately, the resistance to shoulder-surfing was proven false [23].

Cued-recall based schemes.

Cued-recall systems are based on the idea that pictorial help can simplify the task of recall for a user. The prime example is the PassPoint scheme, which goes back to a patent of Blonder [5]. Wiedenbeck et al. [43], [44], [42] and Chiasson et al. [14] studied mostly the usability of the PassPoint scheme. However, a number of later papers showed [20, 31, 36] that the click points are far from being uniformly distributed which substantially weakens the security of the PassPoint scheme. An attempt to point users to more secure passwords can be found in a paper by Chiasson et al. [15].

Suo et al. [34] proposed a variant which is somewhat resistant to shoulder-surfing, further variants include *cued click points* (CCP) [17], where images are changed after each click based on that click, and *persuasive CCP* (PCCP) [15], where while initially selecting the click points the system suggests a specific image region. Estimations for the effective key sizes for PassPoints, CCP, and PCCP are given by Chiasson et al. [16]. Frequent passwords in graphical password schemes have been identified by van Oorschot and Thorpe [37] and we complement this kind of analysis via our user studies.

Under the name Windows 8 Picture Passwords (W8PP), a cued-recall scheme is used in Windows 8 for graphical login. Here, the user can use gestures, a mouse, a stylus, or an alternative input device to enter some password based on his picture. All in all, this gives a wider variety than Android unlock patterns. As Windows 8 is marketed as "one OS for all platforms", we can expect to see W8PP at a variety of systems: For desktops, laptops and tablets, the security of W8PP is clearly higher than any Android login scheme could achieve. Even correlation attacks between the picture and the password drawn [31] do not change this evaluation. Here we exploit that people simply follow the most prominent features of a picture (a.k.a. *points* in Android Unlock Patterns) when drawing their password on the screen. On smartphones, things become different as the display becomes smaller so people cannot draw too many different patterns on the space provided. In a sense, the prominent features attacks make W8PP degenerate to Pass-Go on devices with a very small screen.

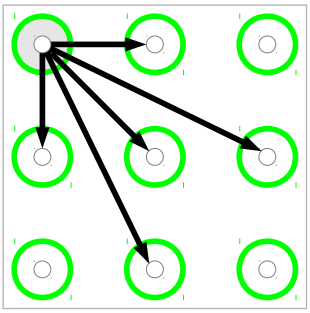


Figure 1: Android's graphical login mechanism and it's reachable points starting from the upper left point.

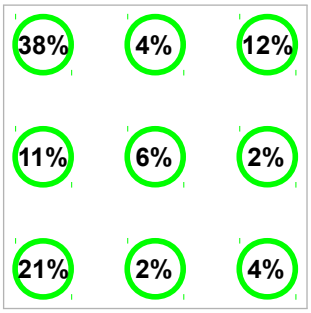


Figure 2: Bias of the initial point in Android Unlock Patterns from the pen-and-paper survey.

3. ANDROID UNLOCK PATTERNS

The unlock patterns used in Android are a simplified version of the Pass-Go scheme. We review the Pass-Go scheme in the following, describe the user study we performed to collect these patterns, and provide basic statistics about the collected data.

3.1 Pass-Go and Android Unlock Patterns

Pass-Go [35] is a graphical password scheme proposed by Tao and Adams in 2008 that can be seen as a successor of the Draw-a-Secret (DAS) scheme [24]. Pass-Go displays a grid of, say, 9×9 dots (similar to a Go board, hence the name). The secret (password) of a user is a drawing on that grid, i. e., a sequence of lines connecting the dots. These lines do not need to be connected, i. e., multiple strokes can be used, and a user may select different colors for each stroke. In order to log in, a user needs to reproduce the secret on a blank grid.

The Android OS uses a simplified variant of the Pass-Go scheme to increase usability and to adapt for the small screens found on typical devices running Android. It uses 9 points arranged in a 3×3 grid, typically on a touchscreen. The user can select a path through these points according to the following rules:

(i) At least four points must be chosen,
(ii) No point can be used twice,
(iii) Only straight lines are allowed, and
(iv) One cannot jump over points not visited before.

The first rule ensure a certain minimal strength of the resulting patterns, albeit little is known about the exact implications on pattern strength. The second and third rules resolve ambiguities from graphical representations of the patterns, which possibly increases usability.

Figure 1 shows the grid and the points that can be reached when starting at the upper left point. One can see that it is not possible to reach the upper right point directly because the straight line contains the upper middle point. Albeit,

the connection between upper left and upper right point is possible, if the upper middle point is chosen previously. There is no easy way to directly calculate the number of possible patterns that follow these rules, but one can easily enumerate all possible patterns and finds that there are $389.112 \approx 2^{19}$ possible patterns, which would be sufficient if users chose their patterns uniformly from this set. It is expected that users do not choose patterns uniformly, and we will quantify this in the remainder of this work.

In addition to use these unlock patterns, Android offers the user the choice of a *personal identification number* (PIN) with four to 16 digits (resulting in a total space of $\sum_{i=4}^{16} 10^i \approx 2^{53}$ PINs) or passwords with four to 16 characters (resulting in a total space of about 2^{107}). For both cases it is well-known that users typically pick PINs and passwords with a distribution that is far from being uniform [9, 28, 40].

3.2 Study Design and Data Collection

As noted previously, there has been substantial work on the usability of graphical password schemes like Pass-Go. However, there is a lack of work in their (practical) security and we aim at closing this gap. One of the main challenges of the work on graphical passwords is collecting reliable user-generated data. In contrast to text-based passwords, where several leaked password lists are available, this is unlikely to happen for Android Unlock Patterns, as they are not stored in a centralized database.

All in all, our aim was to find frequently used passwords. This has to be seen as an initial study on the frequency and security of user passwords in Android Unlock Patterns. When designing this study, we opted for a large amount of data from different people. This is in line with our goal to perform an initial study of the real-world security of Android Unlock Pattern. Hence, usability concerns were *not* the primary focus of our work.

To evaluate the validity of the collected patterns, we additionally conducted a pen-and-paper survey, asking people for basic statistics about their Android Unlock Patterns. Both the user study and the survey were conducted at a large university. For the pen-and-paper survey, we went to different departments on campus and recruited randomly selected people. For all other parts of the study, we recruited university members at the (by far) largest university restaurant. This had the advantage that members of all faculties were represented and that the people were spending 20+ minutes there anyway. The drawback is that we had little to no control over the actual participants.

User study.

For the user study, our main concern was that users choose patterns that are weaker than the ones they would choose on their own mobile phone; note that this is a common concern in user studies. Therefore, we gave each user a "virtual sweet" (i. e., the *promise* that he will receive a real sweet such as for example a chocolate bar *after* the experiment). The task was to "lock" this sweet with a self-chosen, graphical password; we call this a *defensive pattern*.

If the user was able to remember this password after a 20-minute-time-period (in which they typically went for lunch), he could collect his sweet. There was a side-condition, though: No other user participating in our study was supposed to "unlock" this sweet within this 20-minute period. To try to unlock other sweets, each user could (in addition) try up to 5 other patterns (we call these *offensive patterns*).

In summary, the user performs the following tasks:
(i) Chooses a password (defensive pattern),
(i) Attacks up to 5 other passwords (offensive patterns),
(i) Waits 20 minutes (while other users attack this specific sweet), and
(i) Collects his sweet.

By offering users to "attack" other user's sweets we tried to offer an incentive to choose strong passwords. The results of the study (see below) seem to indicate that we achieved this goal, and probably slightly overdid it. A possible explanation is that the chance of losing the sweets was so present when we explained the experiment; for example, all types of sweets were on display as a mean to attract participants.

Giving a full explanation of the psychological background is clearly out of scope for this paper, but based on random, non-structured user feedback there are some explanations.

Looming attack: Interestingly, many people regarded our study as "highly unrealistic" as we assumed that "some random guy has access to my phone for full 20 minutes". So consequently, they chose more robust passwords for their sweets than for their phones as an attack on the first was regarded as far more likely than on the latter.

Environment: Asked by security people in a security study to choose a secure password, people tend to overcompensate and choose rather complicated password. This behavior is well-known in psychology under the name "priming" and documented in several studies [3].

All in all, this leads to the conclusion that the passwords chosen by participants of the study are stronger than the ones we would expect "in the wild".

Pen-and-paper survey.

To check the validity of the results, we conducted a simple pen & paper survey where we asked users for some statistical information about their *real* Android Unlock Pattern. In order to protect the user's patterns we only collected very limited information about those patterns, such as length (number of connected points) and starting point. Furthermore, all collected data was anonymized.

Ethical considerations.

As part of our work, we interviewed 584 people in a user study about their strategies to choose patterns. This user study was performed both as a traditional pen-and-paper survey and also as part of a game. Users were informed before participating in the study that they were to take part in a scientific study and that their data was used to evaluate the strength of Android Unlock Patterns. All data collected during our study was used in an anonymized way (i. e., there was no link between the collected data and an individual user). Furthermore, statistical data was only collected to verify that the overall user sample was not biased.

Our institute does not fall under the jurisdiction of an IRB or similar ethics committee. We did, however, get feedback from peers inside and outside our faculty to validate the ethical perspective of our research.

3.3 Basic Statistics

The pen-and-paper survey was conducted in November 2012. 105 participants helped us by disclosing the starting point and the length of their *real* Android Unlock Pattern.

Table 1: Basic statistic for our first study.

	Male	Female	Total
Engineering Students	35	3	38
Humanities Students	18	8	26
Medicine Students	-	1	1
Science Students	14	5	19
Students (other)	3	-	3
University Employees	15	1	16
Other	8	2	10
Age < 20	23	4	27
Age $21 - 30$	61	15	76
Age $31 - 40$	5	1	6
Age $41 - 50$	-	-	-
Age > 51	4	-	4
Total	93	20	113

Figure 2 shows the bias of the initial point. One can see, that there is a strong bias of the starting point towards the corners. While the probability for all four corners should be 44% in total, we get 75% instead. In contrast to this, the center point, the right, the upper, and the lower center points get only 14% in total. We got 5.63 as average pattern length with a standard deviation of 1.50.

The user study was conducted in the end of 2012 until the beginning of 2013. 113 participants took part in the study. Table 1 summarizes the statistical data on the users we collected while conducting the study. The largest group (38, resp. 34%) of all participants are students of engineering. In addition, 82% of all participants are male and 91% are younger than 31. This data is in line with the overall population of the university, although there is a slight bias towards engineering and male students. The defensive pattern had an average length of 6.59 with a standard deviation of 1.74. For the offensive pattern we obtained 6.33 as average length with 1.68 as standard deviation.

All in all, there is statistical fluctuation between the data of the survey and the study. For example, the bias for the left upper corner is 38% vs. 43%. On the other hand, these statistical differences are in line with the sample sizes of 105 and 113, respectively. However, they do support our claim that the users "in the wild" choose less secure passwords than in our study, e. g., expressed in absolute password length. So the entropies computed in the remainder of this article is more an upper bound than an exact number on the entropy present in Android Unlock Patterns.

We discuss the results at the end of the next section.

4. STRENGTH EVALUATION

In the following, we give a framework to analyze the collected data about Android Unlock patterns regarding their strength. In particular, we need to establish a formal notion of "password strength".

4.1 Brief Introduction to Markov models

The basic idea of Markov models is that subsequent tokens, such as letters in normal text or nodes in the Pass-Go scheme, are rarely independently chosen by humans. For example, in English texts, the letter following a `t` is more likely to be an `h` than a `q`, and for the Pass-Go scheme nodes with distance one are more frequently chosen than distant ones. Based on this observation, in an n-gram Markov model one models the probability of the next token in a string based on a prefix of length $n-1$. Hence, for a given sequence of tokens $c_1, \ldots, c_m$, an n-gram Markov model estimates its probability as

$$\begin{aligned} &P(c_1, \ldots, c_m) \qquad (1)\\ &= P(c_1, \ldots, c_{n-1}) \cdot \prod_{i=n}^{m} P(c_i | c_{i-n+1}, \ldots, c_{i-1}). \end{aligned}$$

In order to use a Markov model we have to determine the *initial probabilities* $P(c_1, \ldots, c_{n-1})$ and the *transition probabilities* $P(c_n | c_1, \ldots, c_{n-1})$, where we get best results when learning these probabilities from data which is as close as possible to the set we are attacking. One can use the relative frequencies of the n-grams to compute the probabilities in the obvious way, but a number of problems arise, e. g., a probability of 0 would be assigned to n-grams that are not part of the training set. Preprocessing of the n-gram counts, so-called *smoothing*, can remedy these problems to a certain extent. We will discuss several design choices regarding Markov models in Section 4.3 and show several experiments that justify our choice of parameters.

4.2 Implementation

When guessing Android Unlock Patterns, or any other authentication strings such as traditional passwords, it is beneficial to guess the candidates in descending order of likelihood. While this is hard for passwords in general due to the large password space, we are in the fortunate position that we can solve this problem in this instance. On a high level, our implementation is structured as follows. The algorithm has access to a *training set* and a *test set*.

(i) We select the *frequent patterns* from the training set, as these are (with high probability) the frequent ones in the test set, too,
(ii) We *learn the n-gram probabilities* from the training set,
(iii) We *estimate probabilities* for all remaining patterns based on the Markov model,
(iv) We sort all patterns in decreasing (estimated) probability,
(v) We evaluate these guesses against the test set.

We perform 5-fold cross-validation on our dataset: we split our dataset into $K = 5$ disjoint subsets $S_1, \ldots, S_5$ of (roughly) equal size; all data points were distributed at random in one of the 5 sets. We select a testing set S_{i_0}, and use the union of the remaining sets as training set. We repeat this process for every set S_i as test set and average the final results over all 10 runs. Our samples have size around 100, so the training sets have size around 80 and the test sets have size about 20.

We use the training set in two different ways: First, we select all patterns that appear more than four times in the training set. While our sample size is too small to guarantee a small approximation error of the relative frequencies, it still provides a hint that it is more frequent than the others. As guessing a few infrequent passwords upfront does less harm than not guessing a frequent one upfront, we use a small threshold of 4. (In preliminary tests we found the results to be insensitive to reasonable parameter choices.)

In order to estimate the frequencies of the remaining patterns, we learn the n-gram frequencies for the Markov model.

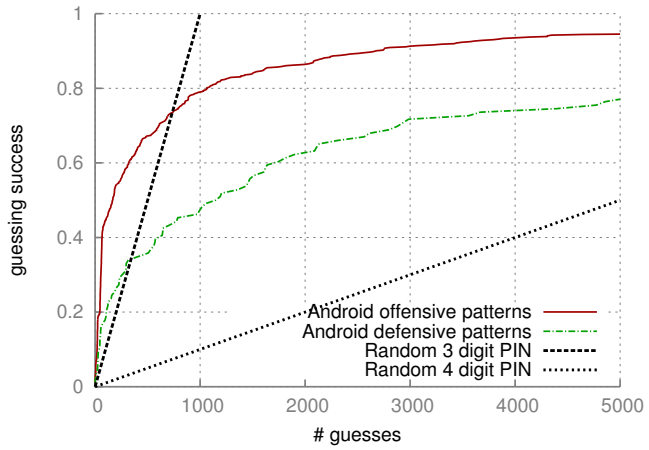


Figure 3: Guessing entropy estimate for plain Android Unlock Patterns.

We use a 3-gram model with a simple Laplace smoothing, see Section 4.3 for experiments justifying these parameter choices. In addition, we use a fixed additional set of patterns which is independent from both the test set and the training set to increase the training material.

Next, we enumerate all valid patterns (i. e., those that fulfill the requirements described in Section 3.1), using a straightforward recursive algorithm. For each pattern, we used both the initial probabilities and the transition probabilities learned before to utility by decreasing probability, overall running time is about five seconds on a standard PC.

Finally, we match the guesses against the test set, recording after how many (simulated) guesses what fraction of the test set was covered.

4.3 Model and Parameter Selection

In this section we discuss some of the design choices we made for the experiments, and discuss the results presented in this section.

3-grams vs. 2-grams.

The choice of the parameter n, i. e., the length of the n-gram, is of crucial importance. In general, longer n-grams yield better results, provided there is a sufficient amount of training data to estimate the occurring probabilities with high enough probability. For $n = 2$ we need to learn (slightly less than) $9 \cdot 8 = 72$ values, for $n = 3$ we need to learn (slightly less than) $9 \cdot 8 \cdot 7 = 504$ values. Each dataset has about 100 patterns at an average length of 6.6. For 2-grams, this yields about 560 data-points or 7.8 data-points per value to learn, for 3-grams, this yields about 460 data-points or 0.9 data-points per value to learn, and for 4-grams, this drops to about 360 data-points or 0.1 data-points per value to learn. This gives a strong indication that we do not have enough data to learn 4-gram probabilities, thus we concentrated on 2 and 3 grams. The results are shown in Figure 4, and clearly 3-grams show better performance. In addition, we believe that the nature of the Android Unlock Pattern scheme suggests the use of 3-grams instead of 2-grams: we found that many users choose points in a straight line, which is not modeled by 2-grams.

Additional information vs. no information.

When collecting data we obtained what we called defensive and offensive datasets (c.f. Section 3.2). We choose the training and test set as subsets of the defensive set. The offensive set is presumably drawn from a weaker distribution, so we can use the offensive set as additional training data. Two contradicting effects come into play: more training data typically means better approximation, but as the additional data is drawn from a (presumably) different distribution, it might actually worsen the results. We tested a number of sizes of the additional dataset, the results are shown in Figure 6. We can see that more data means better results, up to the maximum size of data we have. This is interesting because it seems to hint at the fact that the offensive and defensive patterns have very similar statistical properties.

Smoothing.

When there is not enough data to learn all n-grams, one can use *smoothing* to improve the approximation of n-grams, in particular for the rare n-grams. We tested two different smoothing techniques, simple *Laplace smoothing* (increasing each count by 1) and *Interpolation smoothing* (weighted average between between 2-gram and 3-gram probabilities). We found that the smoothing had very little influence on the success of guessing, which is shown in Figure 5.

4.4 Pattern Strength

A variety of measures for the strength of passwords has been proposed. On a high level, we can distinguish approaches that study *resistance against a specific password cracker* (either by directly attacking them, or by using mathematical models to estimate their effectiveness), and approaches that consider the *distribution of passwords*. While the former are motivated by practice and model real-world attacks pretty well, they highly depend on the specific attack and are usually not generalizable. The latter are based on mathematical models and thus have a clear meaning and are (in some sense) optimal; and usually still provide a reasonable approximation to practical security. See the survey by Bonneau et al. [8] for a review of a number of possible measures.

Partial guessing entropy estimate.

Guessing entropy [12,26] is one metric that can be used to measure the strength of a (password) distribution; it measures the average number of guesses that the optimal attack needs in order to find the correct password. However, an attacker is usually satisfied with breaking a certain fraction of accounts already, which guessing entropy does not take into account. *Partial guessing entropy* [8] (or α-*guesswork*) improves on this.

For $0 \leq \alpha \leq 1$ let $\mu_\alpha = \min\{i_0 \mid \sum_{i=1}^{i_0} p_i \geq \alpha\}$ the minimal number so that the guesses cover at least a fraction α of the passwords, and let $\lambda_\alpha = \lambda_{\mu_\alpha} = \sum_{i=1}^{\mu_\alpha} p_i$ the actual fraction covered (which is greater or equal to α). With these, partial guessing entropy is defined as

$$G_\alpha(X) = (1 - \lambda_\alpha) \cdot \mu_\alpha + \sum_{i=1}^{\mu_\alpha} i \cdot p_i \qquad (2)$$

Here the first term is contributed by those values that weren't guessed in the alloted time, and the second term is contributed by those that were guessed.

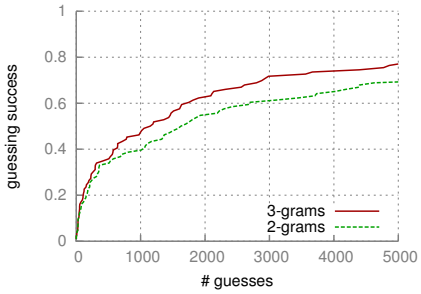


Figure 4: Android (defensive) patterns: Guessing entropy estimate with 3-grams and 2-grams.

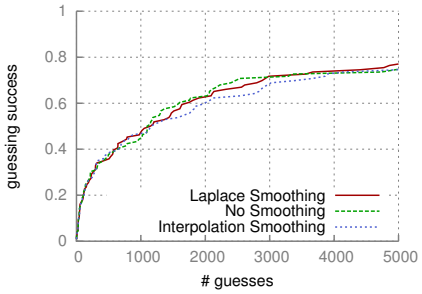


Figure 5: Android (defensive) patterns: Guessing entropy estimate for different smoothing.

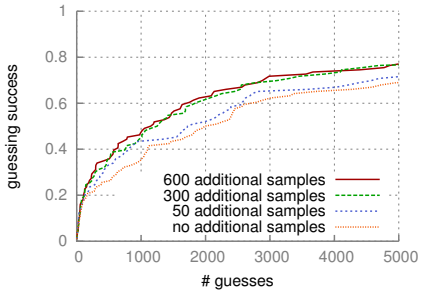


Figure 6: Android (defensive) patterns: Guessing entropy estimate with and without additional information.

Table 2: Comparing partial entropy estimate of several distributions and different values for the target fractions α.

Distribution	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.5$
Android Unlock Patterns (Def, Markov)	8.72	9.10	10.90
Android Unlock Patterns (Off, Markov)	7.56	7.74	8.19
Random Android Unlock Pattern ($U_{389\,112}$)	18.57	18.57	18.57
Random 6-digit PINs ($U_{1\,000\,000}$)	19.93	19.93	19.93
Random 5-digit PINs ($U_{100\,000}$)	16.61	16.61	16.61
Random 4-digit PINs ($U_{10\,000}$)	13.29	13.29	13.29
Random 3-digit PINs ($U_{1\,000}$)	9.97	9.97	9.97
Random 2-digit PINs (U_{100})	6.64	6.64	6.64

We want to express this in "bits of information" to be able to compare it with other measures more easily. This is done as follows:

$$\tilde{G}_\alpha(X) = \log\left(\frac{2 \cdot G_\alpha(X)}{\lambda_\alpha} - 1\right) + \log\frac{1}{2-\lambda_\alpha} \qquad (3)$$

where the term $\log \frac{1}{2-\lambda_\alpha}$ is used to make the metric constant for the uniform distribution (see [8] for a more detailed explanation).

We have two reasons to deviate from this approach. First, to approximate the distribution of X (i. e., the probabilities p_i) requires a certain *size of the sample set* which is beyond the data we collected; second, one can be interested in getting a more *comparable metric for a specific attack*. We are using a combined approach, where we replace the probabilities p_i that are in *optimal order* (as for guessing entropy), with probabilities whose order is given by the actual attack we are considering, i. e., p_i gives the fraction of passwords from the test set that was cracked by the i-th guess. We will refer to this modified guessing entropy estimate simply as *entropy* from here on.

Measuring entropy.

Our entropy estimates are shown in Table 2. We computed (partial) entropy estimates for three levels 10%, 20%, and 50%. We find that these three values span a reasonable range, where breaking half the logins is clearly bad, and even breaking 10% of all accounts is worrisome. As usual for non-uniform distributions, higher values for α give higher entropy estimates. Note that these values are computed on the basis of the attack outlined above, and thus give an upper bound only and the true entropy could be lower. However, in order to exploit such a lower entropy in practice one would need to find an attack that exploits this.

4.5 Evaluation

Results.

The results of our guessing attack is shown in Figure 3, which shows both the success against the defensive and the offensive pattern set. For comparison, we show the respective guessing curves for (randomly-assigned) PINs of three and four (decimal) digits. (4-digit PINs are typically used to protect the SIM-card; user-generated PIN numbers are known to be weaker [9]).

Typically, devices such as mobile phones try to protect against guessing attacks by locking the devices after a number of failed attempts, often requiring a master-PIN to unlock the device. This implies a trade-off between security and usability, which prevents manufacturers from picking too small a number of guesses. With 10 guesses, our data shows that we guess approximately 4% of the accounts correctly for the defensive patterns and approximately 7% for the offensive patterns; with 30 guesses this increases to approximately 9% for the defensive patterns and approximately 19% for the offensive patterns.

Discussion.

The results of our study reveal a lot of shortcomings of the plain Android Unlock Pattern approach. While they provide nearly 400,000 possibilities and are, from a theoretical point of view, therefore more secure than 5-digit randomly-assigned PINs, our evaluations shows that the collected patterns only have an estimated entropy slightly lower than 3-digit randomly-assigned PINs (1.4 bit lower for $\alpha = 0.1$, 1 bit higher for $\alpha = 0.5$). We believe that the collected patterns present an upper bound on the actual strength most Android users lock their smartphone with. This assumption is based on two facts: First, engineers are over-represented in our study and the faculty has a substantial number of students in computer security. Second, the imminent threat of having taken his sweet away, combined with the short timeframe for recall, might have biased the choice of a participant towards a more secure pattern. Furthermore, our evaluation reveals that the "offensive pattern" have a substantially lower estimated entropy, close to the entropy of 2-digit randomly-assigned PINs. We argue that patterns used in the wild have a lower entropy than the collected "defen-

sive pattern" and are more like "offensive pattern" because people without a strong IT security background do not regard a weak pattern as threat for their valuable data on the smartphone. This is also in line with our initial survey on Android Unlock Patterns "in the wild".

5. FINDING MORE SECURE PATTERNS

User choices for the Android unlock patterns are highly biased, yielding security roughly comparable to three digit PINs. We evaluate the influence of several factors on the user choices, in particular we will see that small re-arrangements of the points can yield higher entropy estimates. Secondly, these alternative patterns provide us with insides on the rationals behind user choices.

5.1 Bias Found in the Android Patterns

The design of the Android Unlock Patterns imposes several biases to the patterns chosen by users, that we were able to identify in the collected data. We identified these sources for bias in order to systematically search for better patterns. Here we concentrate on relatively small modifications, in particular re-ordering of the points, because the design of the Android Unlock Patterns is well-known now and has quite good usability. As we will see, relatively small changes are sufficient to substantially improve the security of the scheme, without harming usability.

A basic and obvious weakness is a strong *bias of the starting point* towards the top-left corner, as is shown in Figure 7. While for a uniformly chosen pattern the probability for the top-left corner should be 11%, it is nearly four times higher (44%). Also, we can see a bias towards the corners and away from the center, which is heavily under-represented with 2% probability (less than a fifth from the average). These results are consistent with findings for the PassPoints system [16], which also found a bias to the top-left corner, even though the system is quite different. A plausible explanation is that in the Western hemisphere writing is from left to right and top to bottom.

We see a strong bias towards choosing *adjacent points* (i. e., points with Euclidean distance of 1), and to a lesser extent to *diagonally adjacent points* (i. e., points with a Euclidean distance of 1.4). The first 3-gram that connects two point that are *not* adjacent is the 14-th most frequent 3-gram, and the first 3-gram connecting two points that are not even diagonally adjacent is the 40-th most frequent 3-gram. This is probably caused by problems in usability, as adjacent points are the easiest to draw a connection, and points that are not even diagonally adjacent are very hard to connect without accidentally touching intermediate points. Furthermore, the resulting drawings become quite complicated and probably harder to recall.

There is also a tendency to *stay on the border*, i. e., avoiding the point in the middle. The first 3-gram to touch the middle point is the 12-th most frequent one. Also, there is a tendency for *straight lines* and right angles (the first non-rectangular corner is formed by the 14-th most likely 3-gram). Again, this is similar to the findings for PassPoints [16], where straight lines often can be found.

5.2 Alternative Patterns

We explored four approaches that systematically change the arrangement of the points in the grid without modifying the basic method.

For the first alternative, the *Leftout Small Pattern* (cf. Figure 9), we omitted the upper left point, as it had by far the strongest bias. Our hope was that this would spread the initial point to a larger set of three or four points, thus potentially increasing the entropy of chosen patterns. However, reducing the number of points has the potential to reduce both length and complexity of patterns, thus decreasing entropy.

The second alternative, the *Leftout Large Pattern* (cf. Figure 10), is similar to the previous one, but we added two points in the bottom row to increase the overall point count. We added symmetry by leaving out the bottom right point as well. Most mobile devices use a screen which is larger in one direction, so fitting a 3×4 grid of points should not be problematic.

The third alternative, the *Circle Pattern* (cf. Figure 11), targets another form of bias: For the plain Android Unlock Pattern, there are a many straight lines (horizontal, vertical and diagonal) that users prefer to follow, as we have seen in the previous section. In addition, by removing corner points (which were chosen as starting point in 78% of all patterns, see Figure 7), we hoped to increase entropy. A potential weak point of the *Circle* is that it might be tempting to just "follow the circle" to create a pattern.

The fourth and last alternative, the *Random Pattern* (cf. Figure 12), aims to break all symmetry and thus forcing the user to choose strong patterns. The arrangement of points looks fairly random, with the side condition that no three points should be on a straight line, and no point should be a clear upper-left corner point.

All these patterns were tested in a user study similar to the one before, with 366 participants approximately evenly spread over the four approaches, collecting 2,196 patterns in total.

5.3 Study Setup and Data Collection

We conducted a user study identical to the one described for Android Unlock Patterns in Section 3.2. This allows for a fair comparison of the data we collected.

The user study was conducted over several weeks between the end of 2012 and the beginning of 2013. In total, 366 people participated, some statistics about their affiliation with the university, gender, and age is given in Table 3. The data from the plain Android user study is given to facilitate comparison, it indicates that the population participating in the study should be comparable.

Combining both studies, 479 people supported our work by participating, at least 80 for each approach. The majority of participants are male students of engineering. One explanation is that they have a higher interest in mobile phone security and were therefore easier attracted to participate in our study.

5.4 Evaluation

We used the same methods as explained in Section 4 to test the strength of the collected patterns. The results are shown in Figure 15, and the resulting entropy estimates are given in Table 4.

For the *Leftout Small* patterns, we can indeed observe a more uniform distribution of the first point. However, overall the entropy is lower, most likely due to the smaller number of points and, as a consequence, the lower number of possible patterns.

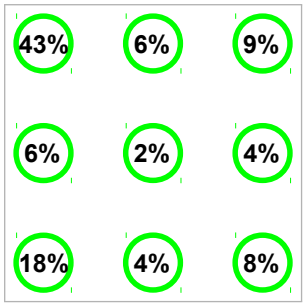


Figure 7: Bias of the initial point.

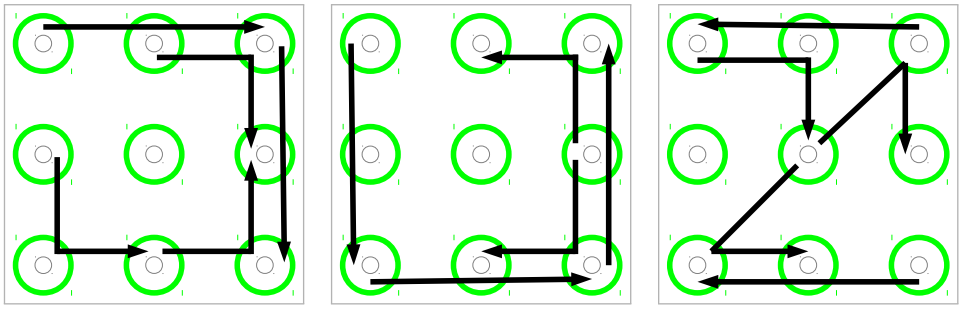


Figure 8: The most frequent 3-grams, from most frequent (left) to less frequent (right).

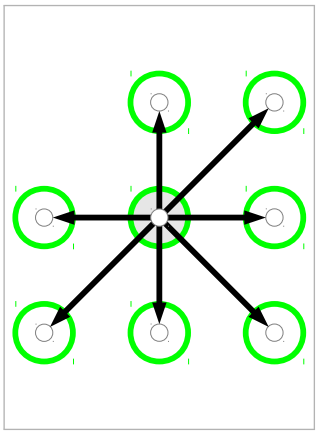


Figure 9: The Leftout Small approach and its reachable point starting from the center.

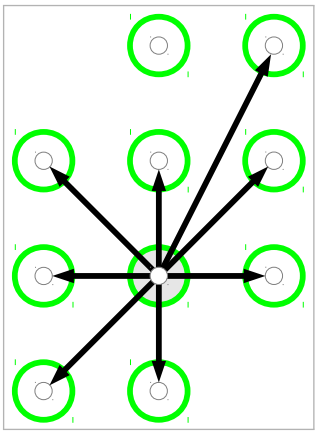


Figure 10: The Leftout Large approach and its reachable points starting from the lower center.

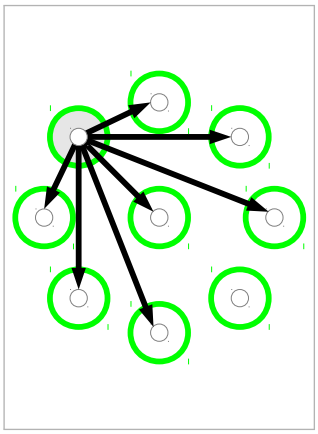


Figure 11: The Circle approach with the upper left corner as starting point and its reachable points.

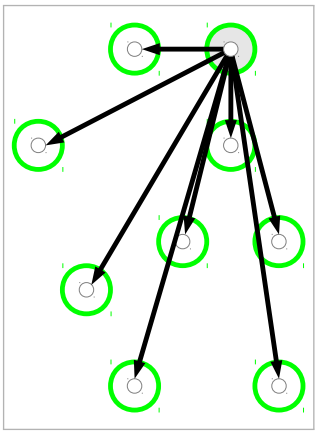
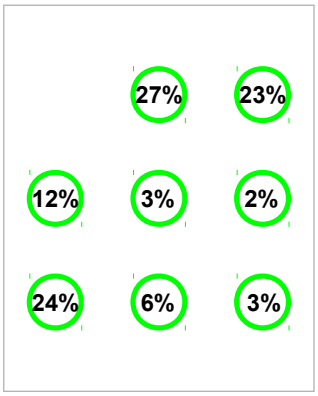
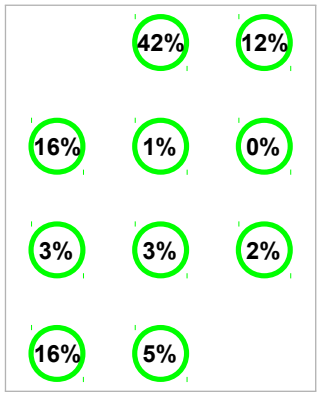
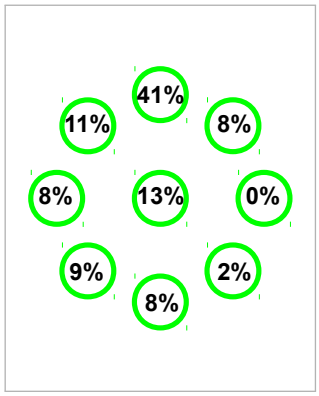


Figure 12: The Random approach with the upper right corner as starting point and its reachable points.







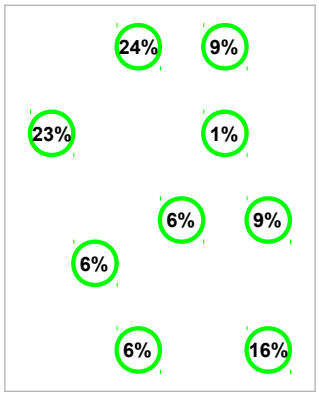
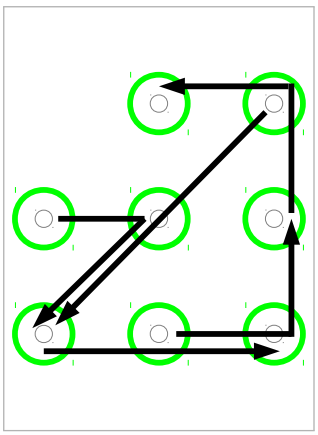
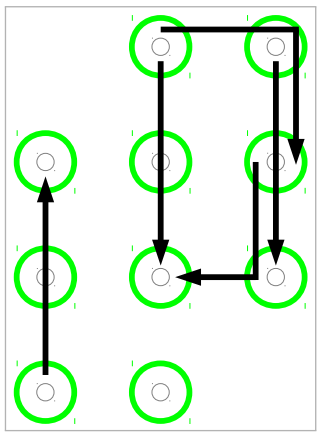
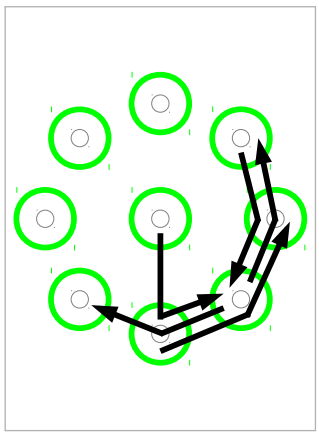


Figure 13: Bias of the initial point for Leftout Small, Leftout Large, Circle, and Random.







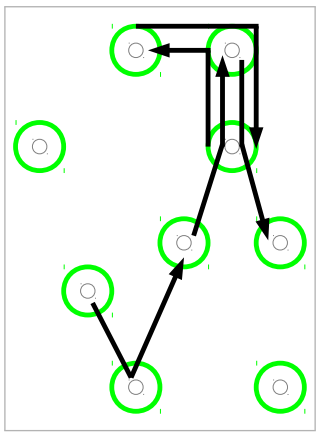


Figure 14: The most frequent 3-grams for Leftout Small, Leftout Large, Circle, and Random.

Table 3: Basic statistics for the plain Android and the alternative pattern user study.

	Plain Android Study	Alternative Pattern Study				
	Total	Total	Lefto. S.	Lefto. L.	Circ.	Rand.
Engineering students	38	133	39	37	31	26
Humanities students	26	74	20	21	18	15
Medicine students	1	39	17	9	4	9
Science students	19	40	9	13	7	11
Students (other)	3	8	6	2	-	-
University Employees	16	64	12	15	19	18
Other	10	8	2	2	3	1
Male	93	264	68	78	61	57
Female	20	102	37	21	21	23
Age < 20	27	65	21	15	20	9
Age $21 - 30$	76	267	75	76	56	60
Age $31 - 40$	6	27	5	6	6	10
Age $41 - 50$	-	3	1	2	-	-
Age > 51	4	4	3	-	-	1
Participants						
w/ successful recall	61	189	59	55	44	31
wo/ successful recall	1	41	8	5	9	19
Recall attempts						
Average	1.28	1.42	1.15	1.49	1.59	1.59
Standard Deviation	0.66	0.79	0.40	0.83	1.10	1.04
Total	113	366	105	99	82	80

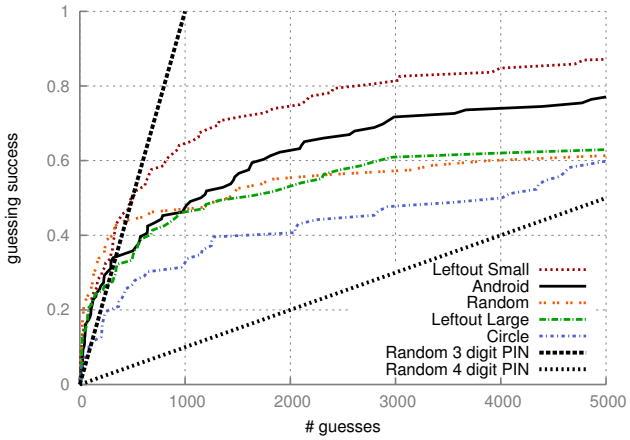


Figure 15: Comparison of the guessing entropy estimates for alternative constructions

For the *Leftout Large* patterns, somewhat surprisingly the bias of the first point is comparable to the original scheme. We do not have a definite explanation, but we believe that the rectangular form of the Leftout Small pattern leads the user to more uniform choices, and the rectangular shape of the Leftout Large patterns emphasizes the asymmetry of the shape. This highlights the importance of user study to understand the actual user choices. Due to the larger number of points, the entropy estimate is larger, and from Figure 15 we can see that the advantage over the original Android scheme is mostly for larger number of guesses.

To our surprise, the *Circle* patterns performed best in our experiments, even though the bias of the initial point is similar to the original patterns. While we had expected that more users would choose to draw a (partial) circle, we saw that many people were aware that simply drawing a circle is very insecure, and resorted to a number of other patterns, like drawing a square or a triangle into the circle. The guessing rate is smaller over the entire range of guesses, and the entropy estimate is about 1.5 bits higher.

The *Random* patterns were surprisingly weak. As expected, the initial point is closer to a uniform distribution. However, even though the points are spread randomly, many users chose a pattern loosely resembling a δ, the reason might be that human try to find patterns even in random data. It is noteworthy that *Random* patterns are particularly weak for very small number of guesses, even weaker than for the original patterns, and get only better for larger numbers of guesses.

Memorability is a clear lack in the above user study: We do not have any data on the long-term development for any given pattern. However, our main focus was to learn about frequently chosen passwords and the rationals behind it, so this question was simply not in focus.

On statistical significance.

The number of samples available for our study is substantially limited by the time requirement for the user study, which raises questions about statistical significance of the results. Unfortunately, guessing entropy does not allow directly computing confidence intervals, therefor we resort to techniques that were previously used by Bonneau [8].

We subsample samples of size 50 from the original dataset of 114 samples, and empirically determined sizes for the confidence intervals from those. We obtain confidence intervals of 8.72 ± 0.38, 9.10 ± 0.65, and 10.90 ± 0.45 for $G_{0.1}$, $G_{0.2}$, and $G_{0.5}$, respectively (for a confidence level of 90%). This indicates that the differences for circle and random are indeed significant even for 50 samples only.

However, this analysis is somewhat problematic, as the sampled subsets are (almost certainly) not disjoint (as 114 samples is not sufficient). But we still believe that it is indicative for the significance of the results, in particular because the above intervals were obtained from 50 samples only, and the larger number of samples should give even smaller intervals.

5.5 Usability Considerations

The main goal of our study was to measure the *security* of the Android Unlock Pattern scheme, as already pointed out in Section 3.2. Furthermore, we introduced small changes in the graphical layout of the scheme to learn about the rationals behind actual user choices for Android Unlock Patterns. Hence, the overall design of the study did *not* take usability into its main focus, but we have collected some data that allows us to argue about the usability of the resulting scheme. First, we recorded the number of attempts people needed to (correctly) enter their passwords. Second, we know how many attempts people needed to enter their password (or fail to do so) after the 20-minute-period. Both give us some indication on the memorability and usability of the scheme. Table 3 compares the usability of all approaches by this metric. The plain Android Unlock Pattern performs slightly better in relation to the *Leftout Large*, *Circle* and *Random* approaches. Only for *Leftout Small*, participants needed less attempts to recall the pattern on average.

A possible explanation is that many participants have already used Android Unlock Patterns before. This is supported by (informal) inquiries during our study. Consequently, they found it easier to choose (and remember) a password in this arrangement of nodes. This is in line with

Table 4: Comparing partial entropy estimate of several distributions and different values for the target fractions α.

Distribution	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.5$
Android Unlock Patterns (Def, Markov)	8.72	9.10	10.90
Android Unlock Patterns (Off, Markov)	7.56	7.74	8.19
Leftout Large (Def, Markov)	7.56	8.73	11.40
Leftout Small (Def, Markov)	8.00	8.93	9.81
Circle (Def, Markov)	9.76	10.81	12.69
Random (Def, Markov)	7.76	7.43	11.10

the fact that *Leftout Small* performed better than all other approaches as has the biggest similarity to the plain approach. In addition, it has the smallest number of nodes. Consequently, the other approaches performed slightly worse as the arrangement of the points was new to the participants. This is particularly true for *Random*. As expected, it performed worst: only 62% of participants could recall their secret within five attempts.

Concluding, our new arrangements are not superior to the plain Android Pattern Login regarding usability. However, we think that this is based in the novelty of the approaches and cannot be seen as shortcoming. It is out of scope of this article to investigate deeper if the usability could be improved, for example by giving the users sufficient training on recalling patterns on the new arrangements.

6. CONCLUSIONS AND FUTURE WORK

To the best of our knowledge, this is the first large-scale user study of Android Unlock Patterns. In particular, we have focused on the *actual* entropy, in contrast to theoretical values such as the size of the key space, that do not provide a reasonable estimate for the strength. Interestingly, around 10% of all users use less than 190 patterns while less than 300 patterns capture around 50% of the whole test popluation (7.56 and 8.19 bit entropy, respectively for offensive patterns, cf. Table 2). Our findings are based on a study with a total of 584 participants. All in all, recruiting these 584 people was rather time consuming and the clear bottleneck of this approach. The dataset is large enough to create Markov models of transitions between points in the 3×3 grid (Android Unlock Patterns) and derive a very efficient attack, yielding a good approximation the exact strength, cf. Table 2 for a detailed comparison of the results for different unlock patterns. However, as explained in Section 3.2, they are most likely upper bounds on the strength we expect to see in real world systems.

To deepen these insights, we tested four simple modifications of Android Unlock Patterns. Interestingly, *Circle* was both a very simple but also a rather secure modification—pointing towards the fact that people like to follow lines (plain Android Unlock Patterns, but also *Leftout Small* and *Large*: As soon as we removed these lines, the overall entropy increased. Apparently, the *Random* pattern was too difficult for users to recognize so they were not able to choose good passwords on the short term. Probably this picture were to change if they were given more time to familiarize themselves with this specific pattern. However, this is out of scope for this article.

We believe some of the ideas presented in our work are worth exploring further. First, it might be possible to choose other patterns (that we did not test) that might offer even better security, at the same level of usability. Our work indicates a couple of potential directions. Clearly, we need to put a strong emphasis on usability here. Second, the system could provide some form of visual feedback to the user about the strength of the pattern he is about to choose: (*red:* weak password, *green:* strong password). While this certainly has the potential to increase the security of the patterns, this must be done more accurately than the prevalent "password rules" that are frequently used for text-based passwords, and it poses questions about usability. Third, we could prevent the attacker from building an attack dictionary (as we did) by having different (challenge) patterns for different smartphones/users.

Acknowledgments.

This work has been supported by the DFG (Emmy Noether grant *Long Term Security* and GRK 187 *UbiCrypt*). We also thank the anonymous reviewers for their valuable insights and comments.

7. REFERENCES

[1] atom. HashCat. Online at `http://hashcat.net/oclhashcat-plus/`.

[2] A. J. Aviv, K. Gibson, E. Mossop, M. Blaze, and J. M. Smith. Smudge Attacks on Smartphone Touch Screens. In *USENIX Workshop on Offensive Technologies (WOOT)*, 2010.

[3] J. A. Bargh, M. Chen, and L. Burrows. Automaticity of Social Behavior: Direct Effects of Trait Construct and Stereotype Priming on Action. *Journal of Personality and Social Psychology*, 71:230–244, 1996.

[4] R. Biddle, S. Chiasson, and P. Van Oorschot. Graphical Passwords: Learning From the First Twelve Years. *ACM Computing Surveys*, 44(4):19:1–19:41, Sept. 2012.

[5] G. Blonder. Graphical Passwords. US Patent 5559961, 1996.

[6] M. Bond. Comments on gridsure authentication. Online at `http://www.cl.cam.ac.uk/~mkb23/research/GridsureComments.pdf`.

[7] J. Bonneau. *Guessing Human-chosen Secrets*. PhD thesis, University of Cambridge, May 2012.

[8] J. Bonneau. The Science of Guessing: Analyzing an Anonymized Corpus of 70 Million Passwords. In *IEEE Symposium on Security and Privacy*, 2012.

[9] J. Bonneau, S. Preibusch, and R. Anderson. A Birthday Present Every Eleven Wallets? The Security of Customer-chosen Banking PINs. In *Financial Cryptography and Data Security (FC)*, 2012.

[10] S. Brostoff, P. Inglesant, and M. A. Sasse. Evaluating the Usability and Security of a Graphical One-time PIN System. In *BCS Interaction Specialist Group Conference (BCS)*, 2010.

[11] S. Brostoff and A. Sasse. Are Passfaces More Usable Than Passwords? A Field Trial Investigation. In *Conference on Human-Computer Interaction (HCI)*, 2000.

[12] C. Cachin. *Entropy Measures and Unconditional Security in Cryptography*. PhD thesis, ETH Zürich, 1997.

[13] C. Castelluccia, M. Dürmuth, and D. Perito. Adaptive Password-Strength Meters from Markov Models. In

Symposium on Network and Distributed System Security (NDSS), 2012.

[14] S. Chiasson, R. Biddle, and P. van Oorschot. A Second Look at the Usability of Click-based Graphical Passwords. In *Symposium on Usable Privacy and Security (SOUPS)*, 2007.

[15] S. Chiasson, A. Forget, R. Biddle, and P. C. van Oorschot. Influencing Users Towards Better Passwords: Persuasive Cued Click-points. In *British HCI Group Annual Conference on People and Computers: Celebrating People and Technology (BCS HCI)*, 2008.

[16] S. Chiasson, A. Forget, R. Biddle, and P. C. van Oorschot. User Interface Design Affects Security: Patterns in Click-based Graphical Passwords. *International Journal of Information Security*, 8(6):387–398, 2009.

[17] S. Chiasson, P. Oorschot, and R. Biddle. Graphical Password Authentication Using Cued Click Points. In *European Symposium on Research in Computer Security (ESORICS)*, 2007.

[18] D. Davis, F. Monrose, and M. K. Reiter. On User Choice in Graphical Password Schemes. In *USENIX Security Symposium*, 2004.

[19] R. Dhamija and A. Perrig. Deja Vu: A User Study Using Images for Authentication. In *USENIX Security Symposium*, 2000.

[20] A. E. Dirik, N. Memon, and J.-C. Birget. Modeling User Choice in the PassPoints Graphical Password Scheme. In *Symposium on Usable Privacy and Security (SOUPS)*, 2007.

[21] P. Dunphy and J. Yan. Do Background Images Improve "Draw a Secret" Graphical Passwords? In *ACM Conference on Computer and Communications Security (CCS)*, 2007.

[22] H. Gao, X. Guo, X. Chen, L. Wang, and X. Liu. YAGP: Yet Another Graphical Password Strategy. In *Annual Computer Security Applications Conference (ACSAC)*, 2008.

[23] P. Golle and D. Wagner. Cryptanalysis of a Cognitive Authentication Scheme (Extended Abstract). In *IEEE Symposium on Security and Privacy*, 2007.

[24] I. Jermyn, A. Mayer, F. Monrose, M. K. Reiter, and A. D. Rubin. The Design and Analysis of Graphical Passwords. In *USENIX Security Symposium*, 1999.

[25] S. Komanduri, R. Shay, P. G. Kelley, M. L. Mazurek, L. Bauer, N. Christin, L. F. Cranor, and S. Egelman. Of Passwords and People: Measuring the Effect of Password-Composition Policies. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2011.

[26] J. Massey. Guessing and Entropy. In *IEEE International Symposium on Information Theory (ISIT)*, 1994.

[27] R. Morris and K. Thompson. Password Security: A Case History. *Communications of the ACM*, 22(11):594–597, 1979.

[28] A. Narayanan and V. Shmatikov. Fast Dictionary Attacks on Passwords Using Time-space Tradeoff. In *ACM Conference on Computer and Communications Security (CCS)*, 2005.

[29] M. Orozco, B. Malek, M. Eid, and A. El Saddik. Haptic-based Sensible Graphical Password. *Proceedings of Virtual Concept*, 2006.

[30] Passfaces Corporation. The Science Behind Passfaces. White paper, available at `http://www.passfaces.com/enterprise/resources/white_papers.htm`.

[31] A. Salehi-Abari, J. Thorpe, and P. van Oorschot. On Purely Automated Attacks and Click-Based Graphical Passwords. In *Annual Computer Security Applications Conference (ACSAC)*, 2008.

[32] Solar Designer. John the Ripper. Online at `http://www.openwall.com/john/`.

[33] L. Standing, J. Conezio, and R. N. Haber. Perception and Memory for Pictures: Single-trial Learning of 2500 Visual Stimuli. *Psychonomic Science*, 19(2):73–74, 1970.

[34] X. Suo. A Design and Analysis of Graphical Password. Master's thesis, College of Arts and Science, Georgia State University, 2006.

[35] H. Tao and C. Adams. Pass-Go: A Proposal to Improve the Usability of Graphical Passwords. *International Journal of Network Security*, 7(2):273–292, 2008.

[36] J. Thorpe and P. C. van Oorschot. Human-seeded Attacks and Exploiting Hot-spots in Graphical Passwords. In *USENIX Security Symposium*, 2007.

[37] P. C. van Oorschot and J. Thorpe. Exploiting Predictability in Click-based Graphical Passwords. *Journal of Computer Security*, 19(4):669–702, 2011.

[38] C. Varenhorst, M. V. Kleek, and L. Rudolph. Passdoodles: A Lightweight Authentication Method. Online at `http://people.csail.mit.edu/emax/public_html/papers/varenhorst.pdf`, 2004.

[39] D. Weinshall. Cognitive Authentication Schemes Safe Against Spyware. In *IEEE Symposium on Security and Privacy*, 2006.

[40] M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek. Password Cracking Using Probabilistic Context-Free Grammars. In *IEEE Symposium on Security and Privacy*, 2009.

[41] R. Weiss and A. De Luca. PassShapes: Utilizing Stroke Based Authentication to Increase Password Memorability. In *Nordic Conference on Human-Computer Interaction (NordiCHI)*, 2008.

[42] S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon. Authentication Using Graphical Passwords: Basic Results. In *Conference on Human-Computer Interaction (HCI)*, 2005.

[43] S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon. Authentication Using Graphical Passwords: Effects of Tolerance and Image Choice. In *Symposium on Usable Privacy and Security (SOUPS)*, 2005.

[44] S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon. PassPoints: Design and Longitudinal Evaluation of a Graphical Password System. *International Journal of Human-Computer Studies*, 63(1-2):102–127, July 2005.