LETTER

A Visibility-Based Upper Bound for Android Unlock Patterns*

Jinwoo LEE†, **Jae Woo SEO**††, **Kookrae CHO**†††, **Pil Joong LEE**†, **Juneyeun KIM**††††, **Seung Hoon CHOI**††††a), *Nonmembers*, *and* **Dae Hyun YUM**††††b), *Member*

SUMMARY The Android pattern unlock is a popular graphical password scheme, where a user is presented a 3×3 grid and required to draw a pattern on the onscreen grid. Each pattern is a sequence of at least four contact points with some restrictions. Theoretically, the security level of unlock patterns is determined by the size of the pattern space. However, the number of possible patterns is only known for 3×3 and 4×4 grids, which was computed by brute-force enumeration. The only mathematical formula for the number of possible patterns is a permutation-based upper bound. In this article, we present an improved upper bound by counting the number of "visible" points that can be directly reached by a point.

key words: *user authentication, graphical password, Android unlock patterns, upper bound*

1. Introduction

To keep strangers from checking out personal information, people often lock their smartphone screens with text-based or graphical passwords. The Android pattern password has been widely used for screen lock technology in Android devices since its first introduction on Android version 1.0. To unlock a locked screen, a user needs to draw a secret pattern that connects a sequence of four or more contact points. For a 3×3 grid, there are 389,112 patterns, which can provide more choices than 5-digit PINs [1]. However, humans do not choose a pattern uniformly at random and have some bias in the pattern selection process, e.g., the upper left corner and three-point long straight lines are very typical selection strategies [2]. Hence, the entropy of patterns is rather low and to increase the security of patterns, schemes with larger grids have been proposed. For example, CyanLockScreen [3] allows users to select from grid sizes ranging from 3×3 to 6×6. The largest grid that we can find in the real life is perhaps the 25×25 grid by Security Lock Screen [4].

Whereas the practical security of unlock patterns should be evaluated by an extensive user study, the theoretical security of unlock patterns can be measured by counting the total number of possible patterns. Currently, it is known that there are 389,112 ($\simeq 2^{19}$) patterns in a 3×3 grid and 4,350,069,823,024 ($\simeq 2^{42}$) patterns in a 4×4 grids [5]. Note that these values were computed by brute-force enumeration because a mathematical formula for the number of patterns is not known even for the simplest case of the 3×3 grid [6]. Can we compute the number of patterns in a large grid (e.g., 10×10) by brute-force enumeration? Even a 5×5 grid is expected to have more than 2^{60} patterns and thus the brute-force enumeration does not seem to be a feasible method for calculating the size of pattern space in a large grid.

The only mathematical formula that can be applied to a large grid is a permutation-based upper bound, which simplifies the counting by ignoring an important restriction on a valid pattern; unvisited points in a valid pattern cannot be jumped over but the permutation-based upper bound allows all kinds of jumps. In this article, we present an improved upper bound by reducing the number of jumps. To refrain from jumping over unvisited points, we use the concept of visibility, i.e., the maximum number of points that are directly reachable from a point. Our numerical data show that the visibility-based upper bound outperforms the permutation-based upper bound.

2. Permutation-Based Upper Bound

In the Android pattern unlock scheme, a user can select a pattern according to the following rules [2]:

(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.

If we use the Cartesian coordinate system where the origin is the point at the lower left corner, a pattern p can be denoted by a sequence of points as $\langle (x_1, y_1), (x_2, y_2), \ldots \rangle$. For example, the Z-shaped pattern in Fig. 1 is the sequence $\langle (0,2), (1,2), (2,2), (1,1), (0,0), (1,0), (2,0) \rangle$.

Suppose the first point of a pattern p in a 3×3 grid is $(0,0)$. Which point can be the second point of the pattern p? If $(0,2)$ becomes the second point, the unvisited point $(0,1)$ must be jumped over and thus, by rule (iv), $(0,2)$ cannot

Manuscript received May 2, 2016.
Manuscript publicized July 25, 2016.
†The authors are with the Department of Electrical Engineering, POSTECH, Pohang, Gyeongbuk, 37673, Republic of Korea.
††The author is with the Software R&D Center, Samsung Electronics Co., Seoul, 06765, Republic of Korea.
†††The author is with the IoT and Robotics Research Division, DGIST, Daegu, 42988, Republic of Korea.
††††The authors are with the Department of Information and Communication Engineering, Myongji University, Yongin, Gyeonggi-do, 17058, Republic of Korea.
*This work was supported by 2016 Research Fund of Myongji University.
a) E-mail: shchoi@mju.ac.kr
b) E-mail: dhyum@mju.ac.kr
DOI: 10.1587/transinf.2016EDL8095

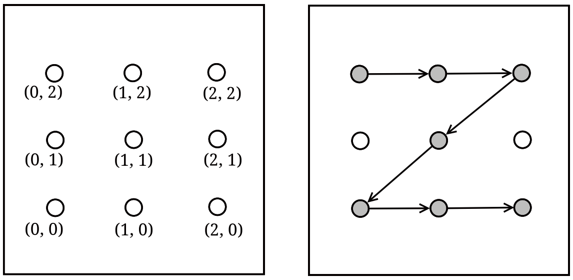

Fig. 1 The Cartesian coordinates and an example pattern for 3×3 grid.

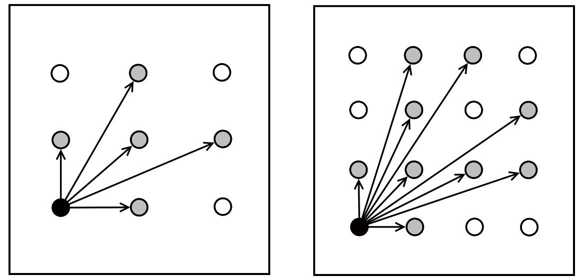

Fig. 2 The visible points from (0,0) in 3×3 and 4×4 grids.

be the second point. Similarly by rule (iv), $(2, 0)$ and $(2, 2)$ cannot be the second point of p. The remaining five points $(1, 0), (0, 1), (1, 1), (2, 1), (1, 2)$ can be the second point of p.

The permutation-based upper bound does not consider rule (iv) and allows jumping over unvisited points; in a 3×3 grid, all eight points except for the first point of a pattern can be the second point and the seven points except for the first and second points can be the third point. Let G be a grid that is a set of points in the plane with integer coordinates. For a grid G, let $|G|$ be the number of points (e.g., for a 3×3 grid G, $|G| = 9$) and $f(G, k)$ be the number of patterns of length k without considering rule (iv). Since any point can be the first point of a pattern, we have

$$\begin{aligned} f(G, k) &= |G| \cdot (|G| - 1) \cdot (|G| - 2) \cdot \cdots \cdot (|G| - k + 1) \\ &= \frac{|G|!}{(|G| - k)!}. \end{aligned} \tag{1}$$

For a grid G, let $\mathcal{N}(G)$ be the number of patterns and $\mathcal{N}_k(G)$ the number of patterns of length k. By rule (i), a pattern should be a sequence of four or more points. Therefore, the permutation-based upper bound on the number of patterns can be obtained as follows:

$$\begin{aligned} \mathcal{N}(G) &= \sum_{k=4}^{|G|} \mathcal{N}_k(G) \\ &\leq \sum_{k=4}^{|G|} f(G, k) \qquad \text{(by ignoring rule (iv))} \\ &= \sum_{k=4}^{|G|} \frac{|G|!}{(|G| - k)!}. \qquad \text{(by Eq. (1))} \end{aligned} \tag{2}$$

The permutation-based upper bound for a 3×3 grid can be found in [6].

3. Visibility-Based Upper Bound

We saw that if $(0, 0)$ is the first point of a pattern p in a 3×3 grid, the number of ways to choose the second point is five, i.e., $(1, 0), (0, 1), (1, 1), (2, 1), (1, 2)$. If $(1, 1)$ is the first point of p, all eight other points can be the second point. Generally, the number of ways to choose the ith point of p depends on the choices of all the previous $i - 1$ points of p. Consequently, the number of ways to choose a pattern of length k grows exponentially in k.

To find an upper bound on the number of patterns in a grid G, we first define the number of points that are directly reachable from a specific point (x, y) as follows:

$$\mathcal{V}((x, y); G) = \left| \{(x', y') \in G \mid (x, y) \rightsquigarrow (x', y')\} \right|, \tag{3}$$

where $(x, y) \rightsquigarrow (x', y')$ means that (x', y') is visible from (x, y), i.e., there is a direct path from (x, y) to (x', y'). The relation $\rightsquigarrow$ satisfies the symmetry property that $(x, y) \rightsquigarrow (x', y')$ if and only if $(x', y') \rightsquigarrow (x, y)$. If the first point of a pattern is (x, y), then the number of ways to choose the second point is $\mathcal{V}((x, y); G)$. Figure 2 depicts $\mathcal{V}((0, 0); G) = 5$ for 3×3 grid G and $\mathcal{V}((0, 0); G) = 9$ for 4×4 grid G.

The visibility $\mathcal{V}(G)$ is the maximum value of $\mathcal{V}((x, y); G)$ over the choice of $(x, y) \in G$. That is,

$$\mathcal{V}(G) = \max_{(x,y) \in G} \mathcal{V}((x, y); G). \tag{4}$$

For any choice of the first point, the number of ways to choose the second point cannot be larger than $\mathcal{V}(G)$. When all $|G| - 1$ points are visible from a point (x, y), $\mathcal{V}$ has the maximum value of $|G| - 1$. Thus, we have

$$\mathcal{V}(G) \leq |G| - 1. \tag{5}$$

After a point (x_i, y_i) is chosen as the ith point of a pattern, it can be jumped over afterwards; hence, the chosen (or visited) point (x_i, y_i) can be regarded as removed from the grid. We define the maximum number of ways to choose the $(i+2)$th point of a pattern, over the choice of the $(i+1)$th point, after i points have been chosen (or removed) as follows:

$$\mathcal{V}(G^{-i}) = \max_{D \subset G, |D| = i} \mathcal{V}(G \backslash D), \tag{6}$$

where $G \backslash D$ is the grid consisting of points in G that are not in D and we have $\mathcal{V}(G^0) = \mathcal{V}(G)$. Since $|G \backslash D| = |G| - i$, Eq. (5) and Eq. (6) give the relation

$$\mathcal{V}(G^{-i}) \leq |G| - i - 1, \tag{7}$$

for $0 \leq i \leq |G| - 2$. Suppose the number of visible points from (x, y) is α, i.e., $\mathcal{V}((x, y); G) = \alpha$. If an invisible point (x', y') is removed from G, the number of visible points from (x, y) does not change, i.e., $\mathcal{V}((x, y); G \backslash \{(x', y')\}) = \alpha$ for $(x, y) \not\rightsquigarrow (x', y')$. If a visible point (x', y') is removed from G, the number of visible points from (x, y) either remains the same or decreases by one, i.e., $\mathcal{V}((x, y); G \backslash \{(x', y')\}) = \alpha$ or $\alpha - 1$ for $(x, y) \rightsquigarrow (x', y')$. For example, Fig. 2 shows

$\mathcal{V}((0,0);G) = 9$ in the 4×4 grid G. For the invisible point $(3,3)$ and the visible points $(1,0)$, $(3,1)$ in Fig. 2, we have $\mathcal{V}((0,0);G\backslash\{(3,3)\}) = 9$, $\mathcal{V}((0,0);G\backslash\{(1,0)\}) = 9$, and $\mathcal{V}((0,0);G\backslash\{(3,1)\}) = 8$. Because removing a (visited) point from a grid does not increase the visibility, we have

$$\mathcal{V}(G^{-i}) \le \mathcal{V}(G^{-(i-1)}) \le \cdots \le \mathcal{V}(G). \tag{8}$$

From Eq. (7) and Eq. (8), $\mathcal{V}(G^{-i})$ satisfies

$$\mathcal{V}(G^{-i}) \le \min(\mathcal{V}(G), |G| - i - 1), \tag{9}$$

where $0 \le i \le |G| - 2$ and $\mathcal{V}(G^0) = \mathcal{V}(G)$.

Since any point can be the first point of a pattern, there are $|G|$ ways to choose the first point. The number of ways to choose the second point is bounded by $\mathcal{V}(G) = \mathcal{V}(G^0)$ and the number of ways to choose the third point is bounded by $\mathcal{V}(G^{-1})$. In general, the number of ways to choose the ith point is bounded by $\mathcal{V}(G^{-(i-2)})$. Now, we can derive the visibility-based upper bound as follows:

$$\begin{aligned}\mathcal{N}(G) &= \sum_{k=4}^{|G|} \mathcal{N}_k(G)\\ &\le \sum_{k=4}^{|G|} |G| \cdot \mathcal{V}(G) \cdot \mathcal{V}(G^{-1}) \cdot \cdots \cdot \mathcal{V}(G^{-(k-2)})\\ &= |G| \sum_{k=4}^{|G|} \mathcal{V}(G) \cdot \mathcal{V}(G^{-1}) \cdot \cdots \cdot \mathcal{V}(G^{-(k-2)})\\ &= |G| \sum_{k=4}^{|G|} \prod_{i=0}^{k-2} \mathcal{V}(G^{-i})\\ &\le |G| \sum_{k=4}^{|G|} \prod_{i=0}^{k-2} \min(\mathcal{V}(G), |G| - i - 1),\end{aligned} \tag{10}$$

where the last inequality is by Eq. (9).

The computation of $\mathcal{V}(G)$ is the most time-consuming part in Eq. (10). Since $\mathcal{V}(G) = \max_{(x,y)\in G} \mathcal{V}((x,y);G)$ is the maximum value $\mathcal{V}((x,y);G)$ over the choice of $(x,y) \in G$, we first compute $\mathcal{V}((x,y);G)$ as follows.

(a) Compute the multiset of slopes $S = \{\frac{y'-y}{x'-x} \mid (x',y') \in G$ and $(x',y') \neq (x,y)\}$ where the slope of any upward (or downward) vertical line is defined by $+\infty$ (or $-\infty$)
(b) Sort the elements of the multiset S and leave only one instance for each slope by removing repeated instances.
(c) Count the (distinct) elements of S in (b).

In step (a), we compute the slopes from (x,y). If multiple points have the same slope, only one point is visible. Therefore, we remove invisible points by removing repeated slopes in (b) and then count the number of visible points in (c). Step (a) and (c) can be done in linear time. Step (b) can be done in $O(|G| \log |G|)$ time by using merge sort or heapsort. $\mathcal{V}(G)$ can be obtained by computing $\mathcal{V}((x,y);G)$ for each $(x,y) \in G$ and finding the maximum value. Therefore, $\mathcal{V}(G)$ can be computed in $O(|G|^2 \log |G|)$ time. In practice, $\mathcal{V}(G)$ for the largest grid of the real life (e.g., 25×25) can be computed less than one second with a desktop PC. Table 1 presents numerical data of upper bounds for up to 25×25 grids. Except for 3×3 grid, the visibility-based upper bound is always lower than the permutation-based upper bound.

Table 1 Upper bound on the number of patterns: visibility-based vs. permutation-based.

Grid	Visibility-based	Permutation-based
3×3	9.86E+05	9.86E+05
4×4	3.60E+13	5.69E+13
5×5	1.48E+25	4.22E+25
6×6	9.24E+40	1.01E+42
7×7	1.08E+62	1.65E+63
8×8	2.99E+87	3.45E+89
9×9	5.60E+118	1.58E+121
10×10	5.68E+154	2.54E+158
11×11	3.13E+197	2.20E+201
12×12	2.18E+244	1.51E+250
13×13	1.84E+299	1.16E+305
14×14	3.45E+358	1.38E+366
15×15	4.74E+425	3.42E+433
16×16	2.61E+497	2.33E+507
17×17	1.86E+577	5.65E+587
18×18	2.50E+661	6.22E+674
19×19	2.47E+755	3.91E+768
20×20	6.90E+852	1.74E+869
21×21	2.20E+960	6.75E+976
22×22	9.17E+1071	2.77E+1091
23×23	6.37E+1192	1.45E+1213
24×24	7.52E+1317	1.16E+1342
25×25	2.55E+1454	1.67E+1478

4. Conclusion

The Android pattern unlock has been widely adopted but very little is known about the mathematical analysis of its security. Except for the "unfinished" work in [6], our upper bound is the first rigorous analysis of the theoretical security of patterns. We invite readers to the research on an exact formula or a tighter bound for the security of patterns.

References

[1] A.J. Aviv, K.L. Gibson, E. Mossop, M. Blaze, and J.M. Smith, "Smudge Attacks on Smartphone Touch Screens," WOOT'10, Proceedings of the 4th USENIX Conference on Offensive Technologies, pp.1–7, USENIX Association, 2010.
[2] S. Uellenbeck, M. Dürmuth, C. Wolf, and T. Holz, "Quantifying the Security of Graphical Passwords: The Case of Android Unlock Patterns," CCS'13, Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, pp.161–172, ACM, 2013.
[3] CyanLockScreen, http://repo.xposed.info/module/com.conceptualideas.cyanlockscreen, 2016.
[4] Security Lock Screen, https://apkpure.com/security-lock-screen/com.lsh.kwj.secure.lock.screen, 2016.
[5] A.J. Aviv, D. Budzitowski, and R. Kuber, "Is Bigger Better? Comparing User-Generated Passwords on 3x3 vs. 4x4 Grid Sizes for Android's Pattern Unlock," ACSAC'15, Proceedings of the 31st Annual Computer Security Applications Conference, pp.301–310, ACM, 2015.
[6] G.C. Kessler, "Technology Corner: Calculating the Number of Android Lock Patterns: An Unfinished Study in Number Theory," JDFSL, vol.8, no.4, pp.57–64, 2013.