LETTER

A Visibility-Based Lower Bound for Android Unlock Patterns*

Jinwoo LEE[†], **Jae Woo SEO**[††], **Kookrae CHO**[†††], **Pil Joong LEE**[†], ***Nonmembers***, ***and*** **Dae Hyun YUM**[††††a)], ***Member***

SUMMARY The Android pattern unlock is a widely adopted graphical password system that requires a user to draw a secret pattern connecting points arranged in a grid. The theoretical security of pattern unlock can be defined by the number of possible patterns. However, only upper bounds of the number of patterns have been known except for 3×3 and 4×4 grids for which the exact number of patterns was found by brute-force enumeration. In this letter, we present the first lower bound by computing the minimum number of visible points from each point in various subgrids.

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

1. Introduction

The Android pattern unlock is a graphical password system that was introduced on Android version 1.0 as an alternative to traditional text-based password systems. A pattern password consists of a sequence of four or more contact points arranged in a grid and the security level of the pattern password is determined by the size of the pattern space. While CyanLockScreen [1] provides grids from 3×3 to 6×6 and Security Lock Screen [2] up to 25×25, the total number of possible patterns is only known for 3×3 and 4×4 grids that was obtained by brute-force enumeration; 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 [3], [4]. The brute-force enumeration is not a candidate method for counting the number of patterns in a large grid because the number of pattern grows too fast.

For grids of large size, only upper bounds are known for the number of patterns: permutation-based upper bound [5] and visibility-based upper bound [6]. Upper bounds enable people to avoid exaggerating the security of pattern unlock but it is a lower bound that guarantees the minimum security level of pattern unlock. To answer questions such as "what size of grid should be used for 2^{80} security level?", a lower bound is required. In this letter, we provide the first lower bound for Android unlock patterns by calculating the minimum visibility of each point in subgrids where already-visited points are removed.

Manuscript received September 22, 2016.
Manuscript revised November 6, 2016.
Manuscript publicized December 1, 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 author is with the Department of Information and Communication Engineering, Myongji University, Yongin, Gyeonggi-do, 17058, Republic of Korea.

*This work was supported by the DGIST R&D Program of the Ministry of Science, ICT and Future Planning (16-IT-04).

a) E-mail: dhyum@mju.ac.kr

DOI: 10.1587/transinf.2016EDL8196

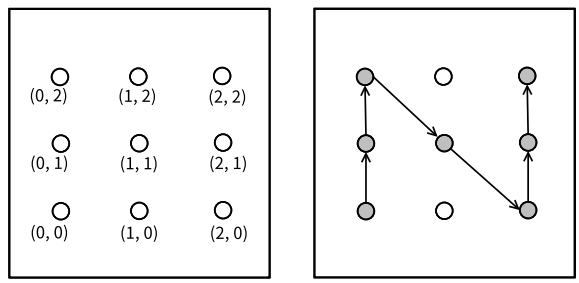

Fig. 1 The Cartesian coordinates and an N-shaped pattern in $G_{3\times3}$.

2. Visibility-Based Lower Bound

2.1 Derivation of Lower Bound

In the Android pattern unlock system, a user can select a secret pattern according to the following rules [7]:

(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.

Let G be a grid that is a set of points in the plane with integer coordinates and $G_{\alpha\times\beta}$ be a grid with α rows and β columns. We use the Cartesian coordinate system to denote a pattern p as a sequence of points $\langle(x_1, y_1), (x_2, y_2), \ldots\rangle$ where the origin is the point at the lower left corner. For example, the N-shaped pattern in Fig. 1 can be expressed by the sequence $\langle(0,0), (0,1), (0,2), (1,1), (2,0), (2,1), (2,2)\rangle$. The number of points in a grid G is denoted by $|G|$. The length of a pattern p is the number of points in p and denoted by $|p|$.

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. Since a pattern should be a sequence of four or more points by rule (i), we have

$$\mathcal{N}(G) = \sum_{k=4}^{|G|} \mathcal{N}_k(G). \tag{1}$$

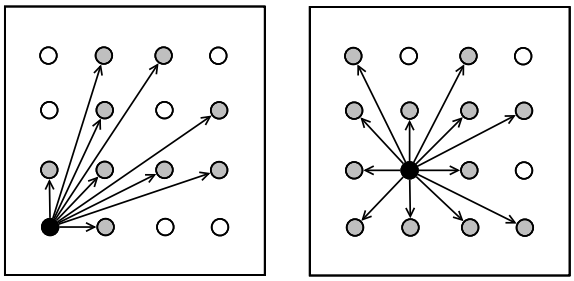

Fig. 2 Visible points from (0,0) and (1,1) in $G_{4\times4}$.

The visibility of a point (x, y) in G is the number of points that are directly reachable:

$$\mathcal{V}((x,y);G) = \begin{cases} \left|\{(x',y') \in G \mid (x,y) \rightsquigarrow (x',y')\}\right| & \text{for } (x,y) \in G \\ \perp \text{ (undefined)} & \text{for } (x,y) \notin G \end{cases} \tag{2}$$

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'). If the first point of a pattern is (x, y), the number of ways to choose the second point is $\mathcal{V}((x, y); G)$. Figure 2 depicts the visible points from $(0, 0)$ and $(1, 1)$ in $G_{4\times4}$, which gives $\mathcal{V}((0, 0); G_{4\times4}) = 9$ and $\mathcal{V}((1, 1); G_{4\times4}) = 12$. The visibility of a point (x, y) that does not belong to G is undefined, i.e., $\mathcal{V}((x, y); G) = \perp$ for $(x, y) \notin G$. The minimum visibility $v(G)$ is the minimum value of $\mathcal{V}((x, y); G)$ over the choice of $(x, y) \in G$. That is,

$$v(G) = \min_{(x,y)\in G} \mathcal{V}((x,y);G). \tag{3}$$

After a point (x_i, y_i) is chosen as the ith point of a pattern, it can be jumped over afterwards (by rule (iv)); hence, the chosen (or visited) point (x_i, y_i) can be regarded as removed from the grid. Let G^{-i} be the set of subgrids of G with $|G^{-i}| = |G| - i$.

$$G^{-i} = \{G\backslash D \mid D \subset G,\ |D| = i\}. \tag{4}$$

where $G\backslash D = G - D$ is the subgrid consisting of points in G that are not in D. We define the minimum visibility of G^{-i} as the minimum 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:

$$v(G^{-i}) = \min_{D\subset G, |D|=i} v(G\backslash D), \tag{5}$$

$$v(G^0) = v(G). \tag{6}$$

The minimum visibility of G^{-i} for a point (x, y) is defined by

$$v((x,y);G^{-i}) = \min_{D\subset G, |D|=i} \mathcal{V}((x,y);G\backslash D). \tag{7}$$

$v(G^{-i})$ of Eq. (5) can be expressed in terms of $v((x, y); G^{-i})$ of Eq. (7).

$$\begin{aligned} v(G^{-i}) &= \min_{D\subset G, |D|=i} v(G\backslash D) \\ &= \min_{D\subset G, |D|=i} \min_{(x,y)\in G\backslash D} \mathcal{V}((x,y);G\backslash D) \\ &= \min_{(x,y)\in G} \min_{D\subset G, |D|=i} \mathcal{V}((x,y);G\backslash D) \\ &= \min_{(x,y)\in G} v((x,y);G^{-i}) \end{aligned} \tag{8}$$

where the third equality follows because $\mathcal{V}((x, y); G\backslash D) = \perp$ for $(x, y) \notin G\backslash D$ by Eq. (2).

Consider patterns $p = \langle (x_1, y_1), \ldots, (x_{i-1}, y_{i-1})\rangle$ and $p' = \langle (x_1, y_1), \ldots, (x_{i-1}, y_{i-1}), (x_i, y_i)\rangle$ in G, where the pattern p' of length i extends p of length $i - 1$. Since a pattern is a sequence of points where order matters, the number of patterns p' that extends p is $\mathcal{V}((x_{i-1}, y_{i-1}); G\backslash D)$ where $D = \{(x_1, y_1), \ldots, (x_{i-2}, y_{i-2})\}$. From $v((x, y); G^{-(i-2)}) = \min_{D\subset G, |D|=i-2} \mathcal{V}((x, y); G\backslash D)$ of Eq. (7), we can get the relation between the number of patterns of length i and that of length $i - 1$ as follows:

$$\mathcal{N}_i(G) \geq \mathcal{N}_{i-1}(G) \cdot v(G^{-(i-2)}), \tag{9}$$

which gives

$$\frac{\mathcal{N}_i(G)}{\mathcal{N}_{i-1}(G)} \geq v(G^{-(i-2)}). \tag{10}$$

Now, we can derive the visibility-based lower bound as follows:

$$\begin{aligned} \mathcal{N}(G) &= \sum_{k=4}^{|G|} \mathcal{N}_k(G) \\ &= \sum_{k=4}^{|G|} \mathcal{N}_1(G) \cdot \frac{\mathcal{N}_2(G)}{\mathcal{N}_1(G)} \cdot \frac{\mathcal{N}_3(G)}{\mathcal{N}_2(G)} \cdots \frac{\mathcal{N}_k(G)}{\mathcal{N}_{k-1}(G)} \\ &\geq \sum_{k=4}^{|G|} |G| \cdot v(G) \cdot v(G^{-1}) \cdots v(G^{-(k-2)}) \\ &= |G| \sum_{k=4}^{|G|} \prod_{i=0}^{k-2} v(G^{-i}) \end{aligned} \tag{11}$$

where the inequality follows from Eq. (10) and $\mathcal{N}_1(G) = |G|$.

2.2 Computation of Lower Bound

To calculate the lower bound of Eq. (11), we need the values $v(G^{-i})$ for $i = 0, 1, \ldots, |G| - 2$. By Eq. (8), we can compute $v(G^{-i}) = \min_{(x,y)\in G} v((x, y); G^{-i})$ by first computing $v((x, y); G^{-i})$ for all $(x, y) \in G$ and then finding the minimum value. We present an algorithm for computing $v((x, y); G^{-i})$ for a point (x, y) and $i = 0, 1, \ldots, |G| - 2$ as follows:

(a) Compute the multiset of angles $S = \{\theta \mid \theta = \text{atan2}(y' - y, x' - x)$ for $(x', y') \in G$ and $(x', y') \neq (x, y)\}$.
(b) Sort the elements of the multiset S.
(c) Count the multiplicity m of each (distinct) element θ of the sorted multiset S in (b) and make a list $L = \langle (\theta_1, m_1), (\theta_2, m_2), \ldots, (\theta_\tau, m_\tau)\rangle$ in order of increasing multiplicity, where $m_j \leq m_{j+1}$ for $j = 1, 2, \ldots, \tau - 1$.
(d) For $i = 0, 1, \ldots, |G| - 2$, set $v((x, y); G^{-i}) = \tau - \ell$ where

$\sum_{j=0}^{\ell} m_j \le i < \sum_{j=0}^{\ell+1} m_j$ with $m_0 = 0$.

In step (a), we compute the angle θ between (x, y) and (x', y') by $\theta = \mathrm{atan2}(y' - y, x' - x)$. In a variety of programming languages, the function $\mathrm{atan2}(\cdot, \cdot)$ is the arctangent function with two arguments [8]. The purpose of using two arguments instead of one is to gather information on the signs of the inputs in order to return the appropriate quadrant of the computed angle. For any real number arguments a and b not both equal to zero, $\mathrm{atan2}(b, a) \in (-\pi, \pi]$ is the angle in radians between the positive X-axis of a plane and the point given by the coordinates (a, b) on it.

Consider $(x, y) = (0, 0)$ in $G_{4\times4}$ (Fig. 2). The angle $\theta = \mathrm{atan2}(y' - y, x' - x)$ is 0 for $(x', y') = (1, 0), (2, 0), (3, 0)$, $\frac{\pi}{4}$ for $(x', y') = (1, 1), (2, 2), (3, 3)$, and $\frac{\pi}{2}$ for $(x', y') = (0, 1), (0, 2), (0, 3)$, which shows that S is a multiset. We sort the angles in step (b) and count the multiplicity of each angle in step (c); the multiplicity m of an element θ in a multiset S is the number of instances of θ in S. For example, $(\theta, m) = (\frac{\pi}{4}, 3)$ indicates that there are three points (x', y') satisfying $\theta = \mathrm{atan2}(y' - y, x' - x) = \frac{\pi}{4}$ for $(x, y) = (0, 0)$. The list $L = \langle(\theta_1, m_1), (\theta_2, m_2), \ldots, (\theta_\tau, m_\tau)\rangle$ is arranged in order of increasing multiplicity, i.e., $m_j \le m_{j+1}$. Since all points $(x', y') \ne (x, y)$ are counted in the list L, the sum of the multiplicities is $|G| - 1$, i.e., $\sum_{j=1}^{\tau} m_j = |G| - 1$.

From the list $L = \langle(\theta_1, m_1), (\theta_2, m_2), \ldots, (\theta_\tau, m_\tau)\rangle$, we can compute $v((x, y); G^{-i})$ for $i = 0, 1, \ldots, |G| - 2$. First, we have $v((x, y); G^0) = \tau$ because the angles of visible points from (x, y) are $\theta_1, \theta_2, \ldots, \theta_\tau$. For G^{-1}, the number of visible points from (x, y) does not change if we remove a point $(x', y') \in G$ with multiplicity $m > 1$ and decreases by one if we remove a point with multiplicity $m = 1$. In general, the number of visible points decreases by one only if we remove m_j points having the same angle θ_j. Since we are interested in the minimum visibility of G^{-i} for (x, y), we reduce the number of visible points as fast as possible by removing points in order of increasing multiplicity. If points of angel θ_1 (with multiplicity m_1) are removed, the visibility becomes $\tau - 1$. If points of angel θ_1 and θ_2 are removed, the visibility becomes $\tau - 2$. Generally, if points of angle θ_j for $j = 1, 2, \ldots, \ell$ are removed, the visibility becomes $\tau - \ell$. Therefore, we have

$$v((x, y); G^{-i}) = \begin{cases} \tau & \text{for } 0 \le i < m_1 \\ \tau - 1 & \text{for } m_1 \le i < m_2 \\ \vdots & \\ \tau - \ell & \text{for } \sum_{j=0}^{\ell} m_j \le i < \sum_{j=0}^{\ell+1} m_j \\ \vdots & \\ 1 & \text{for } \sum_{j=0}^{\tau-1} m_j \le i < \sum_{j=0}^{\tau} m_j \end{cases} \tag{12}$$

where $m_0 = 0$ and $\sum_{j=0}^{\tau} m_j = |G| - 1$.

We can compute $v(G^{-i}) = \min_{(x,y)\in G} v((x, y); G^{-i})$ by finding the minimum value of $v((x, y); G^{-i})$ for $(x, y) \in G$. The visibility-based lower bound can be computed by plugging the values of $v(G^{-i})$ into Eq. (11). Table 1 presents numerical data of the visibility-based lower bound and upper bound, from which the security of Android unlock patterns can be estimated. The total time to compute the lower bound in Table 1 (e.g., $G_{25\times25}$) was less than one second with a desktop PC.

Table 1 Visibility-based bounds for Android unlock patterns

Grid	Lower bound (Eq. (11))	Upper bound [6]
3×3	2.48E+4	9.86E+05
4×4	1.03E+9	3.60E+13
5×5	4.01E+15	1.48E+25
6×6	3.33E+26	9.24E+40
7×7	5.71E+37	1.08E+62
8×8	4.47E+56	2.99E+87
9×9	1.13E+76	5.60E+118
10×10	9.40E+100	5.68E+154
11×11	2.49E+127	3.13E+197
12×12	6.57E+165	2.18E+244
13×13	6.44E+196	1.84E+299
14×14	1.65E+246	3.45E+358
15×15	2.69E+291	4.74E+425
16×16	7.84E+339	2.61E+497
17×17	1.58E+394	1.86E+577
18×18	1.01E+467	2.50E+661
19×19	1.76E+526	2.47E+755
20×20	1.18E+611	6.90E+852
21×21	1.23E+680	2.20E+960
22×22	1.56E+761	9.17E+1071
23×23	1.43E+850	6.37E+1192
24×24	2.56E+961	7.52E+1317
25×25	1.16E+1046	2.55E+1454

2.3 Time-Precision Tradeoff

By spending more computation time, we can improve the lower bound of Eq. (11). If one is willing to compute $\mathcal{N}_\gamma(G)$ where $\gamma \le 3$ (e.g., by brute-force enumeration), Eq. (11) can be rewritten as follows:

$$\begin{aligned} \mathcal{N}(G) &= \sum_{k=4}^{|G|} \mathcal{N}_k(G) \\ &= \sum_{k=4}^{|G|} \mathcal{N}_\gamma(G) \cdot \frac{\mathcal{N}_{\gamma+1}(G)}{\mathcal{N}_\gamma(G)} \cdots \frac{\mathcal{N}_k(G)}{\mathcal{N}_{k-1}(G)} \\ &\ge \sum_{k=4}^{|G|} \mathcal{N}_\gamma(G) \cdot v(G^{-(\gamma-1)}) \cdots v(G^{-(k-2)}) \\ &= \mathcal{N}_\gamma(G) \sum_{k=4}^{|G|} \prod_{i=\gamma-1}^{k-2} v(G^{-i}) \end{aligned} \tag{13}$$

For example, the lower bound for $G_{3\times3}$ can be improved to 4.42E+4 by Eq. (13) with $\mathcal{N}_3(G_{3\times3}) = 320$.

Remark 1. If one is willing to spend even more computing power to calculate $\mathcal{N}_j(G)$ for all $j \in [4, \gamma]$ where $\gamma \ge 4$, then Eq. (13) can be extended to $\mathcal{N}(G) \ge \sum_{j=4}^{\gamma} \mathcal{N}_j(G) + \mathcal{N}_\gamma(G) \sum_{k=\gamma+1}^{|G|} \prod_{i=\gamma-1}^{k-2} v(G^{-i})$. nfortunately, this extension does not seem to be applicable to a large grid because brute-force enumeration is currently the only way to compute

$N_j(G)$.

2.4 Discussion

When we compare $\mathcal{N}(G_{3\times3}) = 389,112$ (or $\mathcal{N}(G_{4\times4}) = 4,350,069,823,024$) with the visibility-based upper bound of [6] and the visibility-based lower bound of Eq. (11), it is the upper bound that is more accurate. This is partially because the event of the minimum visibility seems to happen with a relatively low probability. In step (c) of Sect. 2.2, the list L is computed for each point (x, y) in a grid. For example, consider the list $L = \langle(\theta_1, m_1), \ldots, (\theta_\tau, m_\tau)\rangle = \langle(26.6°, 1), (63.4°, 1), (0°, 2), (45°, 2), (90°, 2)\rangle$ for the point $(x, y) = (0, 0)$ in $G_{3\times3}$ where angles are expressed in degrees. From the list L, we have $v((0, 0); G^0_{3\times3}) = \tau = 5$. To compute the minimum visibility $v((0, 0); G^{-1}_{3\times3})$, we remove the point having the smallest multiplicity, i.e., the point corresponding to $(26.6°, 1)$, which leads to $v((0, 0); G^{-1}_{3\times3}) = 4$. Note that the list L tells us that two angles have multiplicity 1 and three angles have multiplicity 2, i.e., removing one of 2 points corresponding to $\theta = 26.6°, 63.4°$ decreases the visibility but removing one of 6 points corresponding to $\theta = 0°, 45°, 90°$ does not decrease the visibility. Therefore, the probability that the visibility actually decreases is only 25% but we assume this event of 25% probability to compute the minimum visibility. On the contrary, the event of 75% probability is assumed to compute the maximum visibility in [6] and thus the upper bound tends to provide more accurate approximation of the security level.†

Even though the upper bound of [6] is a more accurate approximation (at least for $\mathcal{N}(G_{3\times3})$ and $\mathcal{N}(G_{4\times4})$), it is not known how fast the gap between the upper bound and $\mathcal{N}(G_{n\times n})$ grows as n increases. In addition, people do not choose a pattern uniformly at random and have some bias in the pattern selection process [7]. Therefore, we recommend that one should be as conservative as possible when estimating the security level of the Android pattern unlock; the lower bound of Eq. (11) is less accurate but provides a guaranteed security whereas the upper bound of [6] is more accurate but provides an overestimated security.

Remark 2. In [6], the multiset of angles S was mistakenly defined by $S = \{\frac{y'-y}{x'-x} \mid (x', y') \in G \text{ and } (x', y') \neq (x, y)\}$. Note that $\frac{y'-y}{x'-x}$ cannot distinguish between upward directions and downward directions; e.g., for $(x, y) = (1, 1)$ in $G_{3\times3}$, we have $\frac{y'-y}{x'-x} = 1$ for both $(x', y') = (2, 2)$ and $(x', y') = (0, 0)$. Therefore, the multiset S in [6] should be defined by $S = \{\theta \mid \theta = \text{atan2}(y' - y, x' - x) \text{ for } (x', y') \in G \text{ and } (x', y') \neq (x, y)\}$.

3. Conclusion

Theoretically, the exact security level of the Android pattern unlock system can be computed by counting the number of patterns in a grid. However, a mathematical formula for the exact number of patterns is not known even for the simplest case of the 3×3 grid [5]. Instead, the security level can be roughly estimated by using the lower bound of Eq. (11) and the upper bound of [6]. According to the numerical data in Table 1, there is still a considerable gap between the lower and upper bounds. We invite readers to the research on tighter bounds for the security of unlock patterns.

†We do not claim that the event of the minimum visibility always happens with a smaller probability. We just want to give a possible reason for the large gap between the lower bound and the exact value by explaining typical cases.

Acknowledgements

The authors would like to thank the IEICE reviewer for insightful comments.

References

[1] CyanLockScreen. http://repo.xposed.info/module/com.conceptualideas.cyanlockscreen, 2014.
[2] Security Lock Screen. https://apkpure.com/security-lock-screen/com.lsh.kwj.secure.lock.screen, 2014.
[3] A.J. Aviv, K.L. Gibson, E. Mossop, M. Blaze, and J.M. Smith, "Smudge Attacks on Smartphone Touch Screens," WOOT'10, Proc. 4th USENIX Conference on Offensive Technologies, pp.1–7, 2010.
[4] 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, Proc. 31st Annual Computer Security Applications Conference, pp.301–310, 2015.
[5] 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.
[6] J. Lee, J.W. Seo, K. Cho, P.J. Lee, J. Kim, S.H. Choi, and D.H. Yum, "A Visibility-Based Upper Bound for Android Unlock Patterns," IEICE Trans. Inf. & Syst., vol.E99-D, no.11, pp.2814–2816, 2016.
[7] S. Uellenbeck, M. Dürmuth, C. Wolf, and T. Holz, "Quantifying the Security of Graphical Passwords: The Case of Android Unlock Patterns," CCS'13, Proc. 2013 ACM SIGSAC Conference on Computer & Communications Security, pp.161–172, 2013.
[8] Wikipedia, atan2. https://en.wikipedia.org/wiki/Atan2, 2016.