Under the Table: Tap Authentication for Smartphones

Diogo Marques, Tiago Guerreiro, Luís Duarte, Luís Carriço
LaSIGE, Department of Informatics, University of Lisbon
Campo Grande, 1749-016 Lisbon, Portugal
{dmarques, tjvg, lduarte, lmc}@di.fc.ul.pt

Current smartphone authentication methods are known to be susceptible to even rudimentary attacks based on observation. In this paper, we propose an approach to authentication based on rich tapping patterns that addresses this problem. We present a novel tapping detection technique, using a single example as a template. We also report on two user studies (N = 30 and N = 19) where tapping authentication is compared to the leading alternatives, both in an "out in the open" and in an "under the table" condition. Results indicate that the tapping method approximates current standards of security and usability, but also affords inconspicuous authentication, thus allowing the user to self-protect in social settings.

Tapping, Authentication, Inconspicuous interaction, Smartphones, Detection, Security, Privacy.

1. INTRODUCTION

As our dependence on smartphones and tablets grows, it is easy to forget that in many ways these devices are tokens of our identity. Whoever has my smartphone can send messages to my contacts as if he was me, and access all sort of information about my work and personal life.

Even if we recognize this risk, the cost of mitigating it can be high (Ben-Asher et al. 2011). Typically, these devices will partially power down after a period of inactivity, and an authentication barrier can be set for bringing them back up. Since we use our smartphones in bursts, upon a notification or to accomplish some task, we may have to perform the unlocking procedure many times a day.

It should not be surprising that in the balancing act between security and convenience, the scale so often tilts against security. This is all the more problematic when we consider that increased usage also leads to increased feasibility of some types of attacks. Examples of rudimentary attacks, to which current authentication methods are susceptible, include: (1) smudge attacks, where the attacker narrows down possible code combinations by looking at imprints left on the touch surface; and (2) shoulder-surfing attacks, where a third party is observing us as we authenticate.

In this paper, we present an authentication method that uses tapping sequences on a touch screen. Our Hamming distance-based matching approach accommodates rich tapping patterns of unbounded length, taking into consideration both the times where the user is pressing and the ones in which she releases. Additionally, it tries to match every pattern candidate as the user interacts with the device, thus dispensing timeouts for successful authentication acknowledgment. Furthermore, it is designed in a computationally efficient manner, which allowed us to develop a lightweight tap authentication library for Android smartphones.

The approach allows viable defences against the aforementioned attacks: 1) the screen location in which tapping occurs is irrelevant, thus frustrating smudge attacks; 2) user's visual perception is not needed for tapping, so the authentication task can more easily be performed inconspicuously.

We conducted two user studies: a) out in the open; and b) under the table. Both compared the tap pattern with two leading alternatives: PIN and Android's draw pattern unlock. Two dimensions were addressed: 1) usability; and 2) susceptibility to shoulder surfing attacks.

2. RELATED WORK

Password authentication has been shown to be particularly cumbersome on smartphones. Instead, users tend to favour using a personal identification number (PIN) (Clarke & Furnell 2005). This is the default unlocking option in many smartphones (e.g. iOS, Windows Phone 7.5). To our knowledge, the only widely-used alternative is the Android's draw-pattern unlock. In this gesture-based method, users

Proceedings of BCS HCI – The Internet of Things XXVII, Uxbridge, UK

are presented with a matrix of 9 points, and must input a directed path over them. It is our contention that PIN and draw-pattern unlock are the leading authentication methods in the smartphone world, against which alternative interaction techniques should be measured.

Any recognizer for gesture interaction, can, in principle, be used for authentication. But, as Li (2010) suggests, only very few gesture examples should be needed for subsequent accurate recognition. Thus, data-driven recognizers are more suitable for a mobile context, rather than those relying on parametric properties over a large learning set. Our detection approach follows this prescription, by using very few gesture features, which directly map a single template.

Tap-based authentication has been previously addressed in the literature. Bianchi et al. (2012) propose using taps to insert PINs recurring to the number of touch events. In PassChords (Azenkot et al. 2012), multi-finger touches, using a chord analogy, are used. Unlike ours, these works do not allow unbounded tap pattern lengths, nor take into consideration the up and down time spans, instead being focused on tap counts and location.

In TapSongs (Wobbrock 2009) users can configure tapping patterns representing songs. Although the paradigm is similar, we provide: 1) a simpler implementation not requiring repetition on configuration; 2) the demonstration of similar level of usability and resilience to shoulder-attack in comparison with common alternative methods; 3) the demonstration of a better performance in inconspicuous use regarding those methods. Also, to our knowledge, TapSongs has not been implemented in smartphones or similar devices.

3. TAP PATTERN AUTHENTICATION

Gesture interaction with mobile devices has become common-place. One class of gestures that is widely used is tapping. While key strokes are usually perceived as single events, taps have an implied duration in time. Single taps, long presses and double taps are common examples. These simple patterns can be detected efficiently with very crude algorithms, relying solely on timers.

Tap patterns, however, can be more generally defined as a sequence of intervals of “on” and “off” times – that is, the ordered time distances between and within taps. This definition can be seen as subsuming single-tone music – “on” and “off” intervals are structured into a rhythm. For humans, comparing two pieces of single tone music, or two tap patterns, is a simple enough task, provided some training. For machines, not so much. Indeed, two patterns that a human identifies as being equal, may have a great amount of variation.

We present a technique to compare and match pairs of tap patterns entered by humans. One is defined as a template at configuration time. The other is compared to it at execution time. The output of this comparison is a similarity score, from 0 to 1, where 1 means equality. We implemented this technique as a library for Android applications.

3.1 Tap pattern matching

Starting with an area of the screen that is “at rest”, a tap pattern begins with the first touch event. When the screen is released, the first tap is finished, having lasted an amount of time (which we can label t_1^{down}). The first interval between taps begins. At the next touch event, this interval (t_2^{up}) ends. A tap pattern is then a sequence $\{t_1^{down}, t_2^{up} ...\}$. However, the end of the sequence is uncertain – at the time of the last release event it is impossible to tell if a new release interval has started or if the gesture is finished. For practical purposes, we therefore place an additional constraint on this sequence: it has to start and end with a tap, being of the form $\{t_1^{down}, t_2^{up}, \ldots, t_n^{down}\}$.

3.1.1. Preprocessing

The first step is generating comparable representations of tap patterns that are machine-friendly and still hold enough information for distinctions to be made accurately. Our proposed algorithm first synthesizes a bit array of length 64 from the pattern. This is done by dividing the total time of the sequence (the sum of its members) by 64, obtaining a period. Then, progressing through the sequence, at each period set the corresponding element in the bit array to 0 or 1, depending if it falls within an “on” or “off” member. For example:

$\{200^{down}, 200^{up}, 200^{down}, 200^{up}, 200^{down}\} \rightarrow$
1111111111110000000000001111111111111
0000000000001111111111111

This is an efficient representation, since it consumes very low memory, and allows very fast calculations using logic operators (e.g. OR).

Though it contains information about both the “on” and “off” intervals, certain important features are lost. Returning to our example, any other sequence where the members are symmetrically proportional, such as $\{400^{down}, 400^{up}, 400^{down}, 400^{up}, 400^{down}\}$, will have the same bit array representation. Furthermore, a sequence with a greater number of taps can change the bit array by only a few bits. For instance, the sequence $\{200^{down}, \underline{83^{up}, 34^{down}, 83^{up}}, 200^{down}, \underline{83^{up}, 34^{down}, 83^{up}}, 200^{down}\}$, which contains 5 taps instead of 3, will only change 4 bits. Therefore, our proposed complete representation of a tap sequence is a triplet in the form *(64-bit array, total time, pattern length).* In our example, a pattern would be represented as:

(11111111111110000000000001111111111111
00000000000001111111111111, 1000, 5).

3.1.2. Similarity classification

To compare two tap patterns, we build upon the Hamming (1950) distance, which is widely applied to classification problems. Informally, for two bit arrays, the Hamming distance is simply a count of how many flips would be necessary to make them equal. This distance can be efficiently calculated by applying a XOR operation on the two bit arrays being analysed, and then counting the number of ones in the result. To have a dissimilarity score that varies exactly between 0 and 1, we divide the Hamming distance by the length of the bit arrays. Finally, we subtract the dissimilarity score from one, and have a similarity score. This calculation in only made if 1) the patterns being analysed have the same length and 2) the pattern's total time variation isn't beyond a given proportion (e.g. 20% allowed variation)

3.1.3. Detection at runtime

When a user releases the screen, one of two things might be happening: 1) the pattern insertion was completed 2) a new “off” interval was started. To distinguish these, we use a (configurable) timer to wait for another touch, meaning that “off” intervals will have an upper boundary. But resorting only to a timer implies that detection would have to be delayed, thus increasing the task completion time even for a valid tap pattern insertion.

To overcome this potential hassle, matching is evaluated every time that a pattern candidate emerges, i.e. the user releases the screen. In this way, the timer set when a release occurs only finishes if the patterns do not match – if they matched, that last candidate already had been accepted.

4. STUDY 1: UNLOCKING OUT IN THE OPEN

Two user studies were conducted. The threat model considers that a third party is overlooking as the user, unaware of the fact, unlocks their device. Cases where this might happen include settings like public transportation and social spaces such as bars.

This first study addresses an out in the open setting and comprehends two parallel experiments: one in which participants performed unlocking tasks and other where they simulated shoulder-surfing attack. tasks. Repeated measures designs were used. The independent variable was the unlock method, with three levels: draw pattern, PIN and tap pattern.

This study aims at establishing a baseline for the three methods answering the following question: Do the three evaluated methods provide similar usability and resilience to shoulder-surfing?

Materials

A Google/Samsung Galaxy Nexus with Android 4.1 OS was used. An application for data-gathering was developed, implementing the three unlocking methods: 1) Android's graphic unlock (draw) pattern, 2) PIN and 3) tap pattern. For the tap method, we used the library described in the previous section; for the others we used Android open-source code. The app was instrumented to present a single ease question (SEQ) after each unlocking task (Sauro & Dumas 2009) – a 7-point scale from “very easy” to “very difficult”. The app generated an XML file containing logs of every interaction.

Participants

Thirty students from a local university were recruited: 17 male and 13 female. Ages ranged from 21 to 50 years old, with the average being 26 (SD = 6). Only 2 participants reported not being at all familiar with smartphones; 13 reported being extremely familiar. 11 participants reported currently using either a PIN or password to unlock their personal devices; 8 reported using Android's draw pattern. Participants were offered no compensation.

Procedure

Participants were introduced to the experiments and explained their roles as unsuspecting **user** and opportunistic **observer**. The user was given a smartphone, with the test application already set-up for data collection. For each unlock method users learned or configured their code and tried it out until they were confident that it was memorized.

The observer was then called to observe above the shoulder while the other performed the task one last time, having a maximum of 10 trials to complete it. Upon completion, the application prompted the user to answer the SEQ. The device was then given to the observer, which also had 10 trials to replicate the code. Unlocking methods were presented in random order. Each participant acted one time as the unsuspecting user and one time as the opportunistic observer.

Experimental sessions were conducted in various locations around the university campus, including meeting rooms, offices, bars and sidewalks, as per participant convenience. Although this might introduc greater variability in our measures, it increases ecological validity, i.e. the experiment mimics as much as possible real-life situations.

Random 4-digit PINs were supplied to the user by the application; the length 4 being chosen because it is widely used, as the default option in ATMs, SIM cards, etc. For draw patterns, the application also generated random 5-point patterns; the length 5

being the median of a small-scale (N = 11) survey of colleagues. In pilot runs, users showed great difficulty in replicating random tap patterns; we were therefore forced to let them configure their own, with the limitation that it had to contain at least three touches. We limit biases in memorability by allowing unbounded learning time for PIN and draw patterns. Providing random codes for these two methods may lead to greater resilience to attacks, in comparison to tapping, which, being user-configured, may be more intelligible. Tap patterns, not being bounded in time, also can increase task completion times. These limitations are reasonable in so much as they favour the alternatives against which tap authentication is evaluated.

Measures
In the unlocking experiment, we acquired the SEQ score and the total time it took to complete the task. In the observer experiment, we measured the success within 10 trials. The higher this rate is the lower is the method's resilience to shoulder-surfing.

The task completion times did not follow a normal distribution (as indicated by Shapiro-Wilk tests). Friedman tests were used for analysis, as they were for the SEQ results. For the challenge task, since outcomes are binary, a Cochran's Q test was used. The alpha level was set at 0.05. For post-hoc tests, it was adjusted with the Bonferroni correction.

4.1 Results

4.1.1. Unlock task completion time
The mean task completion times were 2.81s (SD = 1.56s) for draw pattern unlock; 3.82s (SD = 2.60s) for PIN unlock; and 3.73s (SD = 3.10s) for tap pattern unlock. The effect of method on this metric was significant ($\chi^2(2) = 17.267$, $p = .000$). Pairwise, significance was only found between draw pattern and PIN unlock ($Z = -2.437$, $p = .015$, $r = .315$). There was no evidence of a significant effect between tap pattern unlock and both PIN ($Z = -.627$, $p = .530$, $r = .081$) and draw pattern unlock ($Z = -2.273$, $p = .023$, $r = .293$). Using a draw pattern is faster than using PIN; in the remaining comparisons, neither method showed to be less or more time-consuming than the other.

4.1.2. Perceived unlock task ease
The median SEQ score was 7 (IQR = 1) for draw pattern unlock; 7 (IQR = 1) for PIN unlock; and 6 (IQR = 3) for tap pattern unlock. Pairwise, tests showed no significant differences between draw and PIN unlock ($Z = -.690$, $p = .490$, $r = 0.089$), and significant differences between tap and both PIN ($Z = -2.670$, $p = .008$, $r = .345$) and draw unlock ($Z = -2.864$, $p = .004$, $r = .370$). Unlocking was perceived as most difficult when using the tap method.

4.1.3. Shoulder-surfing attack success
In the observer task, subjects were able to successfully replicate another person's code (within 10 trials) 5 out of 30 times when using either draw or tap unlock, and 9 out of 30 times when using PIN unlock. To see if the unlock method had a significant effect on this task's completion rate, we ran a Cochran's Q test, and no statistical significance was found ($\chi^2(2) = 2.462$, $p = .292$). In conclusion, no evidence of method having an effect on resilience to shoulder-surfing attacks was found.

5. STUDY 2: UNLOCKING UNDER THE TABLE

While conducting the first study, our subjects had to be constantly instructed not to conceal the phone from the observer. This suggested that important behaviour might have been overlooked – when we know we are being watched, we may try to authenticate inconspicuously. The technology we currently use isn't, however, designed for this purpose. Tap unlocking, on the other hand, being non-visual, should be better for it. We addressed this possibility by conducting a second study, where users were instructed to conceal the device under a table while authenticating.

Nineteen out of the 30 subjects that participated in the previous study were again recruited. This time, the study consisted of a single experiment, where subjects were asked to configure/learn a pattern, and then unlock the device under a table, away from their sight and the ones of potential over-the-shoulder observers. The research questions are, then, the following: a) of the three methods, which is more usable when there is no visual feedback; and b) for each method, how is usability impacted by unlocking being performed inconspicuously, in comparison to the previous setting?

The same apparatus (device with data-gathering app) was used. The 19 subjects that also participated in this study averaged 27 years of age (SD = 7, range: 21-50). The procedure was similar to the one used in the previous unlock experiments, except for the placement of device away from sight. The observer experiment does not apply. Users were allowed to look at the screen between trials, thus observing if they were successful or not, and reposition themselves for a new trial in the latter case. One additional measure was gathered from the logs: the number of input errors.

5.1 Results

In this section, statistics for both conditions are constrained to the 19 subjects that participated in this study. Therefore, for the visual feedback condition, small changes regarding the metrics presented on the first study are to be expected.

5.1.1. Unlock task completion time

In the condition where visual feedback was available, the mean task completion times were 2.77s (SD = 1.31s) for draw, 4.34s (SD = 3.15s) for PIN, and 4.14s (SD = 3.84s) for tap unlock (Figure 1). Without visual feedback, the mean task completion times were 30.32s (SD = 31.42s) for draw, 43.14s (SD = 32.52s) for PIN, and 6.18s (SD = 9.21s) for tap unlock.

The unlock method had an effect on the time it took to complete the task without visual feedback ($\chi^2(2) = 15.474$, $p = .000$). Pairwise comparisons were conducted and no significant differences were found between PIN and draw unlock ($Z = -1.690$, $p = .091$, $r = .274$). However, between tap and both PIN ($Z = -3.380$, $p = .001$, $r = .548$) and draw unlock ($Z = -2.978$, $p = .003$, $r = .483$) the effect

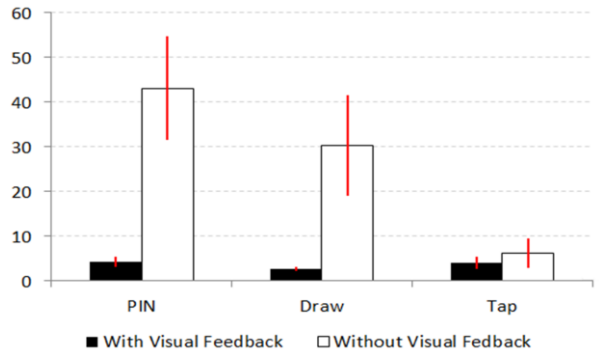

Figure 1. *Mean task completion times (in seconds) for each method and visual condition, 95% confidence.*

was found significant.

Comparing the visual to the non-visual condition for each method, there was no evidence of an effect for tap unlock ($Z = -1.207$, $p = .227$, $r = .196$) There were, however, effects for both draw ($Z = -3.783$, $p = .000$, $r = .614$) and PIN unlock ($Z = -3.823$, $p = .000$, $r = .620$).

We conclude that unlocking without visual feedback is significantly faster using a tap pattern than using a PIN or draw pattern. In this condition, a difference between PIN and draw pattern unlocking could not be established.

5.1.2. Unlock input errors

When there was no visual feedback, the mean number of input errors was 2.84 (SD = 2.544) for draw, 3.53 (SD = 3.325) for PIN, and .42 (SD = 1.387) for tap unlock. Statistical tests showed significant differences ($\chi^2(2) = 15.474$, $p = .000$).

Pairwise, differences between PIN and draw unlock were non-significant ($Z = -0.514$, $p = .607$, $r = .083$). However, again there were significant differences between tap pattern and both between PIN ($Z = -2.894$, $p = .004$, $r = .469$) and draw unlock ($Z = -3.020$, $p = .003$, $r = .490$). We conclude that unlocking without visual feedback is less error-prone when using a tap pattern.

5.1.3. Perceived unlock task ease

We measured the perceived ease of completing the task using the SEQ with equally labelled levels. With visual feedback, the median score was 7 (IQR = 1) for draw, 7 (IQR = 1) for PIN, and 6 (IQR = 3) for tap unlock. Without visual feedback, the median score was 3 (IQR = 2) for draw, 3 (IQR = 3) for PIN, and 7 (IQR = 1) for tap unlock.

When visual feedback was not available, the unlock method had an effect on the subject's perceived ease ($\chi^2(2)=22.377$, $p=.000$). Post-hoc analysis yet again does not show significant differences between draw and PIN unlock ($Z=-1.361$, $p=.174$, $r=0.221$), and shows them between tap unlock and both PIN ($Z=-3.613$, $p=.0000$, $r=.586$) and draw unlock ($Z=-3.536$, $p=.004$, $r=.574$).

The pairwise comparisons between the visual and non-visual settings, do not reveal a significant effect in the case of tap unlock ($Z=-1.342$, $p=.179$, $r=.218$). Such effects were present for draw ($Z=-3.454$, $p=.001$, $r=.560$) and PIN unlock ($Z=-3.742$, $p=.000$, $r=.607$).

We conclude that, without visual feedback, tap unlocking was perceived as the least difficult. Comparing visual and non-visual entry, for tap unlock there's no evidence of a difference, contrary to draw and PIN unlock, which are perceived as more difficult when visual feedback is not available.

6. DISCUSSION

The first user study indicates that tap pattern unlocking is comparable with the two leading methods in terms of usability and resilience to shoulder-surfing. We found no evidence that tap unlock is more time consuming that either PIN or draw unlock. This is a particularly interesting result, since it could not be achieved with a tapping detection solution that did not match patterns whenever a candidate is available. If it were to wait for absolute confirmation that an interaction is finished the method would surely be unusable.

Tap unlock was perceived to be more difficult than the alternatives. It is easy to see why PIN unlock can be perceived as easier – we are accustomed to it and use it in many critical contexts.

The first study also addresses shoulder-surfing resilience. The results showed no statistical significance, but the fact that PINs were successfully replicated by the subject playing the attacker 9 times, in comparison to the 5 observed in the tap and draw pattern unlock methods, should give us pause. This is likely a symptom of a phenomenon we observed while doing the experiments: 4-digit PINs are fast to memorize, at

least for a short period of time; committing a tap or draw pattern to memory takes more time. While the user that was authenticating had unlimited time to memorize the code, the attacker could only do it in the short period she was observing the victim. Anecdotal evidence of this is present in the logs: 1) when the users were given a PIN to learn and try, in almost every case they only did it once; 2) as for draw patterns, there are many instances where users tried the code up to 3 times before indicating to the researcher they were ready for unlocking in the experiment setting (being observed).

The second study confirmed that tap unlocking is an adequate solution for situations where a user wants to do it inconspicuously. For PIN and draw pattern unlock the time it takes and the number of errors greatly increases in this condition. The same cannot be said for tap pattern unlock. The subjective perception of easiness, not surprisingly, is in line with these findings.

One limitation of this study is that it does not address the memorability of tap patterns. We argue, however, that this factor is of much smaller relevance in smartphone authentication methods than in other platforms and technologies. Since we use these devices in bursts, we have to authenticate many times a day, and so memorizing the pattern is made easier. Also, we have observed that the tapping patterns chosen by users are usually mnemonics for something they can easily remember. Examples include, as expected, the rhythm of some piece of music, Morse-like codes, and counts of the number of syllables in a word.

The proposed technique may also be susceptible to other types of attack, namely what is sometimes called “compromising emanations” (for instance, capturing sound or sensor readings), or automated observation through video cameras. Nevertheless, these attacks are complex and also feasible with the other evaluated methods. Tap authentication provides at least some added protection against the casual opportunistic attacker.

7. CONCLUSION

Mobile devices are increasingly seen as extensions of one’s body. They detain information about their owners to such large extent that mobile phone usage can only be sustainable if a high sense of perceived safety endures. Authentication methods present on mobile phones are the tools of the trade for such perception.

However, the currently available procedures for such purposes fall short in protecting the user against the most rudimentary attacks. Moreover, the mobile contexts these devices are used in are prone to present the most challenging (and likely awkward) situations for a secure usage of one’s favourite tool, e.g., authenticating in crowded public transportation.

In this paper, we present a rhythmic approach for mobile authentication that mitigates this mismatch between authentication methods and the mobile context. Results from user studies suggest it is usable in “out in the open” settings. Most importantly, they also indicate that this method enables non-visual usage, creating the required opportunities for inconspicuous authentication.

Future work will consist of non-laboratory validation addressing issues of memorability and environmental factors that may prevent adoption. A security assessment of the tap phrase space and how it relates to the proposed algorithm is being currently conducted.

8. ACKNOWLEDGEMENTS

This work was funded by FCT through project PTDC/EIA-EIA/117058/2010 and the Multiannual Funding Program. Thanks are extended to study participants and anonymous reviewers.

9. REFERENCES

Azenkot, S., Rector, K., Ladner, R., Wobbrock. J. (2012). PassChords: secure multi-touch authentication for blind people. In: *Proc. ASSETS '12*. New York: ACM Press. 159-166.

Ben-Asher, N., Kirschnick, N., Sieger, Meyer, J., Ben-Oved, A., Möller, S. (2011). On the need for different security methods on mobile phones. In: *Proc. MobileHCI '11.* New York: ACM Press. 465-473.

Bianchi, A., Oakley, I., Kwon, D.S. (2012). Counting clicks and beeps: Exploring numerosity based haptic and audio PIN entry. *Interacting with Computers,* 24 (5). 409-422.

Clarke, N.L., Furnell, S.M. (2005). Authentication of users on mobile telephones – A survey of attitudes and practices. *Comp. & Sec.* 24 (7). 519-527.

Hamming, R.W. (1950). Error detecting and error correcting codes. *Bell Tech. J.,* 29. 147-160.

Li, Y. (2010). Protractor: a fast and accurate gesture recognizer. In: *Proc. CHI '10*. New York: ACM Press. 2169-2172.

Sauro, J., Dumas, J.S. (2009). Comparison of three one-question, post-task usability questionnaires. In: *Proc. CHI '09*. ACM Press. 1599-1608.

Wobbrock, J.O. (2009). TapSongs: tapping rhythm-based passwords on a single binary sensor. In: *Proc. UIST '09*. ACM Press. 93-96.