COMPUTERS & SECURITY 66 (2017) 115–128

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

DRAW-A-PIN: Authentication using finger-drawn PIN on touch devices

Toan Van Nguyen [a,*], *Napa Sae-Bae* [b], *Nasir Memon* [a]

[a] *Computer Science and Engineering Department, New York University School of Engineering, Brooklyn, NY, USA*

[b] *Faculty of Science and Technology, Rajamangala University of Technology Suvarnabhumi, Thailand*

ARTICLE INFO

Article history:
Received 11 September 2016
Received in revised form 3 January 2017
Accepted 22 January 2017
Available online 27 January 2017

Keywords:
Gesture authentication
Finger-drawn PIN
Behavioral biometric
Shoulder surfing
Touch devices

ABSTRACT

This paper presents Draw-A-PIN, a user authentication system on a device with a touch interface that supports the use of PINs. In the proposed system, the user is asked to draw her PIN on the touch screen instead of typing it on a keypad. Consequently, Draw-A-PIN could offer better security by utilizing drawing traits or behavioral biometrics as an additional authentication factor beyond just the secrecy of the PIN. In addition, Draw-A-PIN inherently provides acceptability and usability by leveraging user familiarity with PINs. To evaluate the security and usability of the approach, Draw-A-PIN was implemented on Android phones and 3203 legitimate finger-drawn PINs and 4655 forgery samples were collected through an extensive and unsupervised field experiment over 10 consecutive days. Experimental results show that Draw-A-PIN achieves an equal error rate of 4.84% in a scenario where the attacker already knows the PIN by shoulder surfing. Finally, results from a user study based on the System Usability Scale questionnaire confirm that Draw-A-PIN is highly usable.

1. Introduction

The Personal Identification Number (PIN) – or a sequence of digits – remains one of the most prevalent forms of user authentication mechanisms despite numerous efforts to devise alternatives. For example, PINs are used for unlocking mobile devices, authenticating ATM transactions and chip-and-pin card transactions. Bonneau et al. provided a nice analysis of the advantages of PINs based on a Usability–Deployability–Security ("UDS") framework and argued that PINs will be hard to replace (Bonneau et al., 2012).

However, it is known that PINs have several weaknesses. First, PINs are often easy to guess as users typically choose simple, memorable PINs, i.e., sequential digits or birthdays (Bonneau et al., 2012). Second, PINs that are entered by typing on a virtual touch-based keypad are susceptible to a "smudge attack" where the PINs can be learned from the smudges left behind by the user's fingers (Aviv et al., 2010). Third, PIN-based authentication is vulnerable to shoulder surfing in which an attacker obtains a user's PIN by direct observation when in close proximity. More recently, researchers have demonstrated that they can reconstruct a PIN from a low quality video (captured by a smartphone or Google Glass camera from a distance) by recording the hand or fingertip movement in a PIN entry process, with a success rate up to 90% (Shukla et al., 2014; Yue et al., 2014). Given the increased prevalence of handheld recording devices and the ever expanding public surveillance infrastructure, the threat of PIN observation has become stronger than ever.

In this paper, an alternative method to *enter* a PIN on touch devices is proposed. The idea is to let a user draw the PIN on a touch screen instead of typing it on a keypad. Our hypothesis is that the way people draw PINs differs from person to

* *Corresponding author.*
E-mail address: toan.v.nguyen@nyu.edu (T.V. Nguyen).
http://dx.doi.org/10.1016/j.cose.2017.01.008

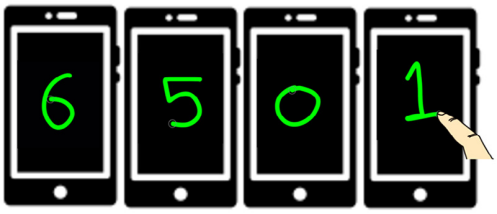

Fig. 1 – Example of Draw-A-PIN on the phone: Instead of typing, each digit in the PIN is drawn on the screen, one at a time.

person, even when they draw the same PIN. Therefore, the system could use both the PIN and drawing behavior characteristics to authenticate a user. This method is called Draw-A-PIN, as it combines the security of a Draw-a-Secret (DAS) (Dunphy and Yan, 2007; Jermyn et al., 1999; Shahzad et al., 2013; Sun et al., 2014) system and the usability of PINs. An illustration of Draw-A-PIN on mobile phones is given in Fig. 1. Draw-A-PIN reduces some security drawbacks of PIN based authentication. First, it can be more resilient against shoulder surfing where an attacker learns the digits that comprise the PIN. This is especially true if the PIN is invisible in the sense that the drawn PIN is not rendered on the touch screen. Second, it provides increased resistance to smudge attacks by requiring each digit in the PIN be drawn on top of previous digits, one at a time. Last, it still provides significant security level even if the user chooses a weak PIN. In addition, Draw-A-PIN offers a *sightless* (eyes-free) feature which allows a user to draw the PIN without looking at the screen. This feature, highlighted in related work (Sherman et al., 2014; Sun et al., 2014), makes Draw-A-PIN applicable in situations where a user wants to authenticate discreetly.

Why draw PINs but not arbitrary curves or even a signature? Draw-A-PIN has an advantage that it does not require users to learn new schemes or new types of secrets and therefore, leverages the high adoption rate of PIN-based systems. In addition, Draw-A-PIN can be built as an option alongside regular PIN authentication as a secure mode which can be invoked when a user needs to authenticate in a public space or is concerned about shoulder surfing. It can also be implemented as a protection method for sensitive applications on the phone, like banking and email apps.

Why do we need Draw-A-PIN *when there are physiological biometrics like fingerprint and iris for authentication?* First, in many legacy systems using PINs like ATM machines, physiological biometrics will not be available in the foreseeable future due to the cost of deployment and privacy concerns (Prabhakar et al., 2003). Second, physiological biometric traits are not secrets and researchers have shown that they can be spoofed (Fatindez-Zanuy, 2004; Matsumoto et al., 2002; Schuckers, 2002; Security Research Labs, 2013). For example, fingerprint authentication on popular Apple and Samsung devices has been easily and quickly broken (Arora et al., 2016; Chaos Computer Club, 2013). Third, even on devices that use physiological biometrics for authentication, PIN is still the main authentication mechanism. For example, on iOS devices, the chosen PIN is part of the encryption mechanism, and thus, a PIN is required whenever the device is restarted (Apple Inc., 2015). Moreover, in these systems, there is usually a fallback authentication method when biometric verification fails, and PIN is usually the fallback. An attacker can bypass biometric authentication by invoking the fallback authentication. Last but not least, users are more and more concerned about their privacy and protection of private data on their devices when dealing with law enforcement and physiological biometrics cannot help them. Specifically, in the US, law enforcement can potentially force users to give their biometrics like fingerprints to unlock their devices but cannot force users to reveal their PIN or passcode due to the Fifth Amendment (Geuss, 2014; Kravets, 2015). One inevitable downside of Draw-A-PIN, like any biometric systems, is that enrollment is required to establish biometric templates or references for verification. To shorten enrollment time, Draw-A-PIN only asks users to provide a small number of drawn PINs, e.g., three to five samples.

The main contributions of this work are as follows.

- We propose Draw-A-PIN, an eyes-free, two-factor, and shoulder surfing resistant PIN-based authentication system using finger-drawn digit interaction. Specifically, we demonstrate that, in addition to the secrecy of the PIN itself, finger-drawn digits can be utilized as a second factor for user authentication since they carry drawing traits that are typical to individuals.
- We develop a finger-drawn digit PIN authentication algorithm comprising two cascade modules: PIN Content Analyzer and Drawing Behaviour Analyzer to verify the two factors of a log-in attempt.
- We evaluate Draw-A-PIN in different settings through extensive and unsupervised experiments. Specifically, we conduct two studies to evaluate the effectiveness of Draw-A-PIN system under two threat models where the attacker has different levels of knowledge about user's finger-drawn PIN. In addition, we explore the template aging problem and propose a potential solution for it by means of a template updating strategy.

The rest of the paper is organized as follows. Related works are surveyed in Section 2. Draw-A-PIN system overview and threat model it has been designed for are presented in Section 3. System components are detailed in Section 4 and Section 5. An evaluation of Draw-A-PIN performance through unsupervised field experiments is detailed in Section 6. Limitations and discussion are presented in Section 7. Finally, this paper is concluded in Section 8.

2. Related work

In this section, approaches that have been proposed as alternatives to PINs for user authentication are discussed and related to our work. These can be divided into two categories: gesture-based and biometric authentication.

2.1. *Gesture-based authentication*

One approach to authenticate users is the Draw-a-Secret (DAS) scheme and its variants (Dunphy and Yan, 2007; Jermyn et al.,

1999; Shahzad et al., 2013; Sun et al., 2014). These approaches exploit the increasing popularity of the touch interface present in most modern devices. They allow the user to draw a memorizable pattern on a device's screen in order to authenticate. DAS systems require users to learn and utilize a new type of security mechanism. This often results in a high learning curve as a user has to understand what a complex or unpredictable pattern is. For example in the Android locking/unlocking scheme, the DAS system was demonstrated to be vulnerable to the same attacks PINs fall victim to (Serwadda and Phoha, 2013). That is, users pick common and predictable patterns. Draw-A-PIN does not require users to learn or adapt with any new secret. Users only switch from typing to drawing digits, which is a natural task to almost everyone.

Recently, GEAT, proposed by Shahzad et al., authenticates a user based on how she inputs specific gestures on a touch display prompted at enrollment and verification (Shahzad et al., 2013). Sherman et al. relaxes the constraint by allowing the user to draw free-form gestures for authentication (Sherman et al., 2014). TouchIn allows a user to unlock the phone by drawing free-form single or multiple finger gestures on an arbitrary region of the display without looking at it (Sun et al., 2014). Sae-Bae et al. proposed five-finger gestures on multi-touch devices (Sae-Bae et al., 2012, 2014). Although these authentication schemes have potential, they have similar drawbacks as many proposed DAS systems. Again, unlike Draw-A-PIN, they require the user to learn a new type of secret.

De Luca et al. proposed improving the security of Android locking patterns by adding an additional security layer to the patterns (De Luca et al., 2012). Users are authenticated by how they input the pattern and the correct pattern. Specifically, users are verified based on a set of features including pressure, size, and speed of drawing. However, this scheme has a high false acceptance rate of 21% at a threshold where the false rejection rate is minimized at 19%. The data set includes 26 participants with 645 valid login samples and using every other participant's samples as attack samples for each participant.

Online handwritten signature can also be used for user authentication on mobile devices (Mendaza-Ormaza et al., 2011; Sae-Bae and Memon, 2014). In such a system, a user draws her signature on a touch screen (with finger or stylus), which is then compared against a template stored on the device. While the user may not need to learn a new secret, false positives or false negatives can be high depending on the complexity of and consistency of user signatures (Sae-Bae and Memon, 2015).

3D gestures have recently been proposed for authentication (Liu et al., 2009a, 2009b). Bailador et al. (2011) proposed a system where a user draws her signature in the air while holding her phone, and the signature is captured by the 3D accelerometer sensor available on modern mobile devices. This approach is not only vulnerable to shoulder surfing but also can cause social awkwardness. Similarly, the system described in KinWrite (Tian et al., 2013) is also a 3D handwritten gesture authentication, but it requires an external device.

2.2. *Biometric authentication*

Biometric authentication that uses physiological or behavioral characteristics to authenticate a user can be resilient to shoulder surfing. Systems using physiological characteristics such as fingerprints, iris, and face for authentication have been explored for many years. These approaches are believed to be more secure and are a convenient way to lock/unlock mobile devices. Examples are Apple TouchID and Samsung Fingerprint Scanner. However, there are several concerns regarding these approaches. First, typical physiological biometrics are usually publicly available, i.e. users leave their fingerprints on anything they touch or their faces are frequently captured by surveillance cameras, so attackers can obtain them. For example, Apple TouchID and Samsung Fingerprint Scanner were shown to be broken using reconstructed fingerprint of users left on phone case or a glass surface (Cao and Jain, 2016; Chaos Computer Club, 2013; Whitney, 2014). Although this reconstruction might not be so easy, it has been improved recently by Cao and Jain (2016). Specifically, they generated 2D fingerprint spoofs in under 15 minutes using a victim's fingerprint image, conductive ink and a AgIC special paper, that could be used to break into a victim's mobile phone protected by fingerprint authentication. Second, as demonstrated by researchers, these biometric traits can be spoofed (Biggio et al., 2012; Cao and Jain, 2016; Fatindez-Zanuy, 2004; Gupta et al., 2014; Matsumoto et al., 2002; Schuckers, 2002; Security Research Labs, 2013). Last, physiological characteristics are considered permanent and irreplaceable and can be used for tracking users across systems, and hence raise privacy concerns (Prabhakar et al., 2003).

On the other hand, systems using behavioral characteristics such as keystroke patterns (Feng et al., 2013; Maiorana et al., 2011), or touch dynamics (De Luca et al., 2012; Frank et al., 2013; Sae-Bae et al., 2012; Shahzad et al., 2013; Sherman et al., 2014; Sun et al., 2014) are considered less intrusive and are attracting attention from researchers as a potential replacement for passwords. In addition, it has been recommended by researchers to combine biometrics with a form of authentication using a secret (Riley, 2006). Draw-A-PIN is an example of this type of system where PIN drawing traits or drawing behavior of the user can be used as an additional authentication factor.

3. Draw-A-PIN system overview

In this section, the elements of touchscreen data utilized by Draw-A-PIN are presented first. Then, an overview of Draw-A-PIN system and the assumed threat model are provided.

Touch screens are used on most phones, tablets, and wearable devices today. Each time a user draws a curve with her finger on a touchscreen, a sequence of touch events are generated which can be retrieved through the OS. Each touch event includes a set of attributes, among which the following are of interest to this work: *x,y-coordinate* of the touch point, *pressure* indicates how hard the finger was pressed on the screen, *size* of touch area, and a *timestamp* of the touch event. Values of x–y coordinates are in the range of the screen resolution. For example, x is from 0 to 1080 and y is from 0 to 1920 on a Nexus 5 phone. Pressure and size have the values in the range of 0 and 1.

As depicted in Fig. 2, Draw-A-PIN consists of an enrollment phase and an authentication phase.

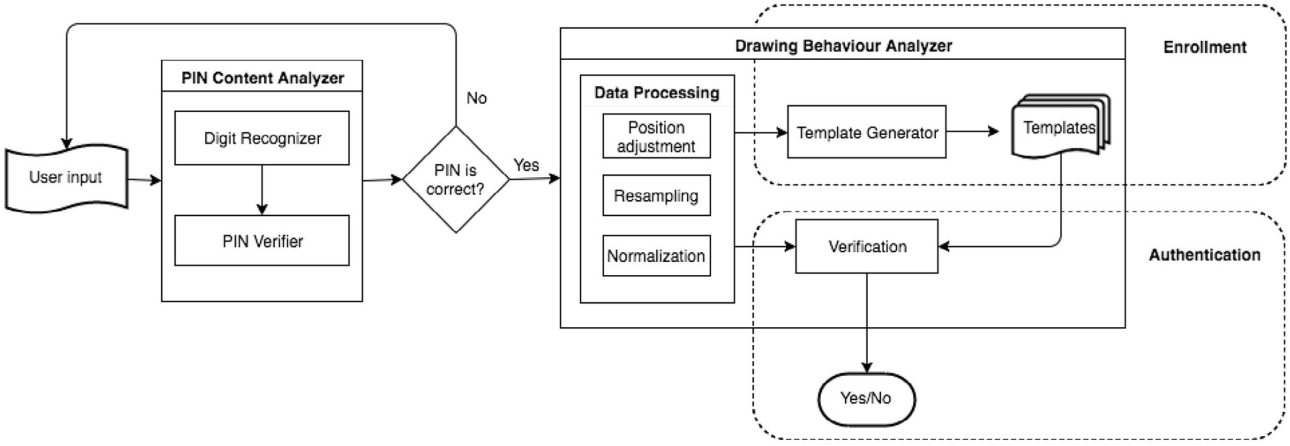

Fig. 2 – Draw-A-PIN system overview.

- *Enrollment:* In this phase, a user is prompted to choose a PIN and is asked to draw this PIN a small number of times (for example 3 to 5 times) on a touch sensitive display. The system then extracts information from each input sample including x-coordinates, y-coordinates, finger pressure, and size of touch area. Draw-A-PIN processes all entered samples and stores them on the device as the authentication template for this user.
- *Authentication:* In the authentication phase, anyone attempting to unlock the device needs to draw a PIN on the display. When a PIN is drawn, the system first recognizes the individual digits and checks the PIN; if it is the correct PIN, the system then verifies the drawing behavior of the PIN by calculating a distance between the entered sample and the stored authentication template using a pattern matching algorithm. If this distance is below a predefined threshold, access is granted to the user.

Draw-A-PIN requires the user to draw each digit of the PIN, one by one, such that when a digit is drawn, its smudge wipes out the trace of previous digits. Thus, Draw-A-PIN potentially provides some resilience against a smudge attack.

In terms of the system components, all drawn digits are sent to the *PIN Content Analyzer* (PCA) (detailed in Section 5). The PCA checks whether the user has entered the correct number of digits of the PIN, passes each digit sample to a *Digit Recognizer*, and then verifies the correctness of the PIN. If the PIN is incorrect, then the user is asked to enter the PIN again. Otherwise, the PCA sends the PIN to the *Drawing Behaviour Analyzer* (DBA) (detailed in Section 4) to verify user's drawing characteristics. The DBA accepts or denies this sample based on its similarity with the stored template. As a result, the system achieves two-factor authentication but requires only one step from the user. The first authentication factor is the PIN, the secret. The second authentication factor is the extracted features that correspond to the user's behavioral and physiological characteristics. They are significantly difficult for attackers to imitate even if he knows the PIN as well as access to some drawing sequence samples, as shown by evaluations in Section 6.

3.1. *Threat model*

There are two types of threats that are considered for Draw-A-PIN. The first type, which is often not considered in the literature, is called the "*curious insider threat*". By this it means that a family member or a colleague who has numerous opportunities to observe a PIN being entered, and consequently ends up learning it without deliberate and malicious intention. In a study by Muslukhov et al. (2013), it was found that users are concerned about insiders (defined in the paper as people who are familiar with users, e.g., friends, co-workers) accessing their data on smartphones and they recommended that a stronger adversarial model that incorporates the insider threat should be considered during the design and evaluation of authentication methods for smartphones.

To evaluate the realistic nature of the curious insider threat we conducted an online survey of 267 mobile users (62% were male and their ages ranged from 18 to 60), that composed of questions about their phone locking behavior, locking mechanisms, awareness about threats, and their concern about the curious insider threat in comparison with other threats. Interestingly, 194 participants (73%) reported that they have observed and learned someone else's PIN. In addition, 210 participants (79%) knew a friend or a family member's PIN. The distinction between these two cases is that in the first case, participants had observed and learned someone's PIN while the second case is restricted to family members or friends and they may have observed the PIN or may have been told the PIN value. Also, in the first case, participants may have observed the PIN of strangers (maybe not intentionally, i.e., on a subway). In the situation when devices were not with the owners, 70% of participants thought their family members would probably look at their data, and 75% thought their friends or colleagues would likely peek into data on their phones. They were less worried about strangers and law enforcement accessing their data in the same scenario as only 63% and 67% worried about these attackers respectively. Moreover, while 83% of participants locked their phones, 59% still used PINs which confirmed the popularity of PINs among mobile users. These

results also show that the threat model is realistic and the users actually care about the curious insider threat.

The second type of threat considered is a targeted attack, where a resourceful attacker goes to great lengths to observe a PIN. For example, ATM skimmers usually install a hidden camera on or near the ATM machine that points to the keypad to record the PIN of the customers (Feinberg, 2014). In addition, as discussed in Section 1, with the increased popularity of handheld and wearable devices equipped with cameras, it is practical for an attacker to record a PIN entry of a user and later recover the PIN as demonstrated in numerous studies (Balzarotti et al., 2008; Maggi et al., 2011; Shukla et al., 2014; Yue et al., 2014).

In the context of the above threats, this paper evaluates performance of DRAW-A-PIN under two different attacks where an attacker has different levels of knowledge about the user's finger-drawn PIN as summarized below:

- **PIN Attack.** This corresponds to the curious insider threat where the attacker knows the PIN of the user, but does not know in any significant detail how the PIN was drawn. Furthermore, the attacker could physically access the user's device and try to impersonate the user to log into the device. An example of this type of attack could be a curious co-worker who intentionally or unintentionally observes a user entering the PIN once or several times, and learns the PIN. Later, when the user goes out and leaves the device on the table, this co-worker tries to log into the device to sneak access to user's data.
- **Imitation Attack.** This corresponds to the targeted threat involving a powerful attacker. Not only does he know the PIN but has access to a video recording of how the user draws the PIN. He trains himself as much as he wants and then imitates the user.

4. Drawing Behavior Analyzer (DBA)

One important aspect of DRAW-A-PIN when compared to regular PIN-based is the ability to distinguish users based on their PIN drawing characteristics. This section describes how DRAW-A-PIN verifies users' behavior by detailing its Drawing Behaviour Analyzer (DBA) component. In addition, a pilot study to demonstrate the applicability of the idea is also presented. Accordingly, PIN Content Analyzer (PCA) component will be described in the subsequent section.

The DBA component consists of *Data Processing* and *Verification* modules.

4.1. Data processing

The Data Processing module preprocesses and prepares the PIN for verification using three steps: 1) Position translation, 2) Resampling, and 3) Normalization.

Let $P = \{d_i\}_{i=1}^{l}$ be a finger-drawn PIN sample in which l is the number of digits in the PIN (PIN's length) and $d_i = \{X_i, Y_i, P_i, S_i, T_i\}$ is a digit. Each digit d_i contains 5 time series with a length of n_i as follows.

x-coordinate: $X_i = \{x_j\}_{j=1}^{n_i}$

y-coordinate: $Y_i = \{y_j\}_{j=1}^{n_i}$

pressure: $P_i = \{p_j\}_{j=1}^{n_i}$

size of touch area: $S_i = \{s_j\}_{j=1}^{n_i}$

timestamp: $T_i = \{t_j\}_{j=1}^{n_i}$

In the first step, each digit is translated to the origin of a Cartesian coordinate system: $x_j = x_j - x_1$ and $y_j = y_j - y_1$, $j = 1, \ldots, n_i$. This step allows a user to draw digits in any region of the screen without looking at it (eyes-free interaction).

Then, in the second step, each digit is resampled to a uniform sampling rate using linear interpolation to accommodate the variations caused by different computational resources on the device when the PIN is drawn. For example, when many background processes are running on the device, the sampling rate may be reduced.

In the third step, values of x,y-coordinates of each digit are normalized to the same scale with pressure and size attributes so that no attribute will dominate the distance used in verification process. Min–max normalization is used in DRAW-A-PIN as follows. For every digit d_i:

$$\bar{x}_j = \frac{x_j - min\{x_j\}_{j=1}^{n_i}}{max\{x_j\}_{j=1}^{n_i} - min\{x_j\}_{j=1}^{n_i}} \quad (1)$$

$$\bar{y}_j = \frac{y_j - min\{y_j\}_{j=1}^{n_i}}{max\{y_j\}_{j=1}^{n_i} - min\{y_j\}_{j=1}^{n_i}} \quad (2)$$

The final feature set includes normalized x,y-coordinates which represent the shapes of the digits and in some sense the speed of drawing (normalized by timestamp), pressure and size of touch represent users physiological characteristics, i.e., how big is the finger, and also represent users behavior, i.e., how hard the finger presses on the screen and how much skin touches the screen which are often unique to each user.

In the enrollment phase, template generator constructs the digit templates using $\bar{x}_j$, $\bar{y}_j$, p_j, and s_j, derived from enrolled samples. Then, in the authentication phase, the digit templates are used along with those derived from a log-in sample to derive a verification decision.

4.2. Drawing behavior verification for finger-drawn PIN

In the enrollment phase, a user is required to draw his or her PIN a small number of times, resulting in a set of authentication templates T: $T = \{T_i\}$. Verification of an entered PIN login sample P is based on a dissimilarity score $Score(P, T)$ which shows how dissimilar P is from T. $Score(P, T)$ is calculated using distance between P and T ($Distance(P, T)$) and median of pairwise distances between samples T_i in T. Numerous approaches to derive $Distance(P, T)$ including the one that combines or averages the samples in T in a single template were tried. The approach that defines $Distance(P, T)$ as the distance between

P and a representative T_{ref} of T which is called the *reference sample* gives DRAW-A-PIN the best performance and is chosen. The details of each step are then presented below.

4.2.1. Determine a reference sample t_{ref}

For each authentication template T_i in T, average pairwise distance between T_i and the rest in T is first calculated. Specifically, the PIN pairwise distance between two PIN samples is computed using the digit pairwise distance between digits of the same PIN digit position from the two PIN samples. Then, the T_i with smallest average distance is selected to be the reference sample T_{ref} for the user. T_{ref} is then used when computing dissimilarity score of a login sample P with respect to the authentication template set T. Details on how to compute pairwise distance between any pair of digits and entered PIN are described below.

- *Digit pairwise distance:* To compare two finger-drawn digit samples, Dynamic Time Warping (DTW) (Sakoe and Chiba, 1978) is selected and applied, as it computes an elastic distance function that can be used to compare two time-series signals with unequal lengths. It has been shown to give good performances in related work (Sae-Bae et al., 2012, 2014). Let d_1 and d_2 be the representations of two processed finger-drawn digits with lengths n_1 and n_2. That is, $d_1 = \{\delta_1^{d_1}, \delta_2^{d_1}, \ldots, \delta_{n_1}^{d_1}\}$ and $d_2 = \{\delta_1^{d_2}, \delta_2^{d_2}, \ldots, \delta_{n_2}^{d_2}\}$, where each digit is a series of touch points $\delta_i^* = \left(\bar{x}_i^*, \bar{y}_i^*, p_i^*, s_i^*\right)$.

The DTW algorithm constructs a $n_1 \times n_2$ distance matrix, in which the value of each cell (i, j) is computed as,

$$cell(i, j) = c\left(\delta_i^{d_1}, \delta_j^{d_2}\right) + \min\begin{cases} cell(i-1, j) \\ cell(i, j-1) \\ cell(i-1, j-1) \end{cases} \quad (3)$$

where $c\left(\delta_i^{d_1}, \delta_j^{d_2}\right)$ is a Euclidean distance between two points on the plane.

$$c\left(\delta_i^{d_1}, \delta_j^{d_2}\right) = \sqrt{\left(\bar{x}_j^{d_2} - \bar{x}_i^{d_1}\right)^2 + \left(\bar{y}_j^{d_2} - \bar{y}_i^{d_1}\right)^2 + \left(p_j^{d_2} - p_i^{d_1}\right)^2 + \left(s_j^{d_2} - s_i^{d_1}\right)^2} \quad (4)$$

DTW finds the warping path that best matches these two digits and returns the warping distance that is the minimum cumulative distance of the path in the cell $(n_1, n_2) : W(d_1, d_2) = cell(n_1, n_2)$. The pairwise distance between two digits is calculated as

$$D(d_1, d_2) = \frac{W(d_1, d_2)}{\min(n_1, n_2)} \quad (5)$$

- *PIN pairwise distance:* The pairwise distance between two finger-drawn samples P_1 and P_2 of a PIN is calculated as the sum of pairwise distances between corresponding digits in the two samples as follows.

$$Distance(P_1, P_2) = \sum_{i=1}^{l} D\left(digit_{i(P_1)}, digit_{i(P_2)}\right) \quad (6)$$

where l is the PIN length. When choosing the reference sample T_{ref} for the user, the T_i with smallest average PIN pairwise distance is selected.

4.2.2. Derive a dissimilarity score

To compute a dissimilarity score between a login sample and the template set, all distances between the reference sample T_{ref} and remaining samples in authentication template set T are calculated. There are various ways to derive a dissimilarity score from these distances, as this is a common approach used in related work (De Luca et al., 2012). In the proposed system, we found (empirically from our experiment in Section 6) that a dissimilarity score computed by normalizing the distance between a login sample P and the reference sample T_{ref} with the median of distances between T_{ref} and remaining samples in T gives DRAW-A-PIN its best performance. That is:

$$Score(P, T) = \frac{Distance(P, T_{ref})}{median(Distance(T_{ref}, T_i))} \quad (7)$$

where T is the authentication template set, P is a login sample, T_{ref} is the reference sample, and $T_i \in T$.

4.2.3. Verify a login sample

A login sample is accepted if its dissimilarity score is below a verification threshold. Otherwise, it will be rejected. Typically, in a biometric system, the optimal threshold can be empirically found by varying the threshold value and deriving a Receiver Operating Characteristic (ROC) curve and a Equal Error Rate (EER). False Positive Rate (FPR) and False Negative Rate (FNR) are calculated at each threshold value. True Positive Rate (TPR), which is 1-FNR, and FPR are used to draw the ROC curve. EER is the value where FPR and FNR are equal. In biometric systems, EER is a commonly used as a performance metric to compare the accuracy of different systems. The lower the EER, the better the system performance.

4.3. Pilot study

To determine whether or not drawing characteristics and finger-drawn PINs can be used for user verification, a pilot study for DRAW-A-PIN was conducted in which only the Drawing Behaviour Analyzer component was considered.

A prototype of DRAW-A-PIN was implemented as a web application using HTML5 and PHP to collect finger-drawn PIN samples. Forty participants, mostly students, were recruited for this study. Each participant was assigned a 4-digit PIN so that a balanced distribution of digit samples from 0 to 9 could be collected in the final data set for analysis. This data set would give a better understanding on the performance of DBA on each individual digit as well as on PINs. Each participant was asked to visit the project website and provide up to 6 finger-drawn samples of the PIN with time intervals between each session varying from 12 to 96 hours. This setup resulted in a set of $40 \times 6 = 240$ legitimate finger-drawn PIN samples. At the time of this study, reliable pressure and size of touch events could not be captured from HTML5 API, thus, the data of a drawn PIN only included x–y coordinates and timestamps and were sent to the server once the participant clicked to save his or

Table 1 – The verification performance (EER) when using 5 training samples in cross-validation test.

	PIN attack	Imitation attack
EER (%)	6.70	9.89

her sample. Imitation samples were collected for two attacks in our threat model from 5 attackers who were all students. In the PIN attack, attackers were given a page containing printed PINs of users and asked to draw each PIN in their own style. In the Imitation attack, each attacker was given an animation of the exact reproduction of how the PIN was drawn by a user. He studied the animation for 20 s and then imitated the user. This was repeated for all users. Each attacker provided 6 imitating samples of each user for each attack scenario, which resulted in a set of $5 \times 6 \times 40 = 1200$ samples for each attack.

Due to the relatively small number of legitimate samples, a cross-validation test was performed in which any 5 samples of a participant were used as the enrollment set. The remaining legitimate sample and all imitating samples of the participant were used for testing in each round and in each attack. The results of this evaluation are presented in Table 1 in term of Equal Error Rate (EER) of the system. It can be seen that even when the attackers already know the PINs, Draw-A-PIN still rejects them with a probability of 92.30% and 90.11% in the PIN attack and Imitation attack, respectively, while a legitimate user is falsely rejected 6.7% and 9.89%, respectively. That is, if a user, for example, logins 30 times per day, she will be rejected 2.01 times.

This result confirms that drawing behavior can be leveraged for user verification and DTW can provide Draw-A-PIN reasonable performance as a baseline. This result was reported in a short position paper in Nguyen et al. (2014).

5. PIN Content Analyzer (PCA)

The PIN Content Analyzer is responsible for verifying the correctness of the entered finger-drawn PIN. It consists of a Digit Recognizer and a PIN Verifier module. Based on recognition results received from the Digit Recognizer, the PIN Verifier accepts an entered finger-drawn PIN if it has the right digits in the right order and rejects otherwise. If the entered PIN is accepted, PIN Verifier passes it to the DBA component for behavior verification. In the rest of this section, the Digit Recognizer is described in more detail.

Typically, there are two types of digit recognition systems: off-line and online systems. In an off-line system, an image of the digit is acquired and then processed with image processing techniques before being recognized. In an online system like Draw-A-PIN, a sequence of x–y coordinates with time labels is acquired and used for recognition. While significant amount of research has been done on off-line systems (Cardoso and Wichert, 2013; Ciresan et al., 2012; LeCun et al., 1995; Liu et al., 2003), little research has been done for online systems (Connell and Jain, 2001; Kim and Sin, 2014).

In this work, $P algorithm (Point-Cloud Recognizer), a fast, simple, and accurate gesture recognition approach that is based on templates and nearest-neighbor classification (Vatavu et al., 2012) is adopted and modified to be used as the Digit Recognizer. In $P, each gesture is represented by an unordered point-cloud which allows $P to handle both unistroke- and multistroke-gestures. $P uses Euclidean distance and implements in its Cloud-Distance as the scoring function (Vatavu et al., 2012). To recognize a drawing stroke d, $P calculates distances between d and all samples d_i in the training set and returns the name of the sample with smallest distance as recognition result.

However, the original $P algorithm was designed to recognize custom drawing strokes and some recognition issues arise when it is applied to recognize the drawn digits in a PIN. First, one specific digit drawn by one user could be similar to another digit drawn by different users. For example, the drawn digit "1" of user A could be similar to the drawn digit "7" by user B. Second, the original $P uses a special case of k-NN classifier with $k = 1$, resulting in the nearest neighbor algorithm. This does not necessarily give the best performance as it is shown in Section 6.

Therefore, the original $P is modified to improve the recognition performance when it is applied to Draw-A-PIN as follows.

5.1. *Training and customizing $P for each user*

As described above, a training set *DT* consisting of finger-drawn digits samples from 0 to 9 is a prerequisite. To create *DT*, five volunteers were recruited. Each volunteer was asked to draw 10 digits (0 to 9) on a Google Nexus phone. Drawing each digit five times resulted in a collection of 250 finger-drawn digit samples. Bigger size of *DT* increases recognition time as $P has to compare all samples with an entered digit. Thus, our decision is to randomly choose a sample of each digit from each volunteer to add to *DT*. This resulted in a set of 50 digit samples for the training set.

To cope with the issue where one digit may be drawn in different ways by different users, $P is customized for each user instead of using a generic $P for Draw-A-PIN. Specifically, as described in Section 3.1, each user is required to draw some PIN samples for enrollment. The digits from these PIN samples are added to *DT* of the user. This means that each user has his or her own $P recognizer which matches an input digit with the core 50 digit samples and user's digits from enrollment.

5.2. *Modifying $P algorithm for recognizing finger-drawn digits*

The $P algorithm is modified to improve its performance in recognizing drawn digits and our changes are described in Algorithm 1. Instead of returning the name of the sample with smallest distance, Modified-$P returns the name of the digit that has the most occurrences among k samples that have smallest distances to the input digit d, varying k results in different recognition rates. The best empirical k is presented in the evaluation section. This modification results in **1.8%** increase in average accuracy of recognizing digits and **6%** increase in overall PCA performance (see Section 6.2).

Algorithm 1 Modified $P algorithm to recognize drawn digits

Require: d is an input sample to be recognized, DT is digits training set, k is the number of nearest neighbors

1: **function** MODIFIED-$P ($d, DT, k$)
2: $n \leftarrow 32$ ▷ Resampling length
3: $distances[i].\delta \leftarrow \infty$, $i = 1..len(DT)$
4: NORMALIZE($d.points, n$)
5: **for** $i \leftarrow 1$ to $len(DT)$ **do**
6: $distances[i].\delta =$ CLOUD-DISTANCE($d.points, DT(i).points$)
7: $distances[i].name = DT(i).name$
8: **end for**
9: SORT($distances$) ▷ Ascending based on distance δ
10: $match =$ FIND-MAJORITY($distances[1..k]$) ▷ Find the most popular template in k nearest neighbors. If there is a tie, return the name of sample with smallest distance.
11: **return** $match.name$
12: **end function**

For digit recognition purposes, only x–y coordinates are needed as inputs for the Digit Recognizer. Also, all samples were resampled to have 32 points in length in order to achieve desired performance and execution time as recommended by Vatavu et al. (2012).

6. Main study

A study was conducted to evaluate DRAW-A-PIN comprehensively. Evaluation was done based on a dataset where the PINs were drawn when the screen was set to vertical mode and each digit was drawn one by one on the entire screen as depicted in Fig. 1. The data were also collected in a more realistic situation. The section begins by introducing the study and data collection procedure. Next, an experimental evaluation of the security and usability of DRAW-A-PIN is presented.

6.1. *Study procedure and data collection*

Experiments were designed to simulate everyday use of finger-drawn PINs and DRAW-A-PIN was implemented as an Android app to collect PIN samples. Participant recruitment posting was distributed through our institutional mailing list and Facebook. At the end, 20 volunteers (14 males and 6 females), whose ages were from 20 to 30 years and each owned an Android phone, were recruited as users for our study. These volunteers were different from earlier volunteers (in Section 5 and from the pilot study). At the beginning, the users were introduced to the study and were asked to download the app to their phones. Then each user was assigned a 4-digit PIN so that a balanced distribution of digits can be collected for analysis. When the users first used the app, they were asked to enroll by drawing their assigned PIN numbers five times. Then, they were required to provide login samples through the app every day, at different times and places. This requirement was emphasized in the beginning and was confirmed with users at the end of the experiment that they indeed followed the instructions. Places reported by users include classroom, subway, parks, their home. Notification is sent via the app to each user to remind them to use the app everyday during the experiment. All samples were recorded by the app and sent to a server. Data were collected over 10 consecutive days without supervision. At the end of the experiment, each user filled out a survey about their perception of the security and usability of DRAW-A-PIN.

In the study, two types of finger-drawn PIN were collected. The first was a Visible finger-drawn PIN, which was drawn with visual feedback on the screen every log-in time. The second was the Invisible finger-drawn PIN, which was drawn without any visual feedback once in every 5 logins during the experiment.

In total, 3203 finger-drawn PINs including 100 training samples (five for each user) and 3103 login samples (155.6 per user on average, range 50–307), in which 599 samples were Invisible finger-drawn PINs were collected. Note that Invisible finger-drawn PIN could improve resilience against shoulder surfers but it could also affect authentication accuracy and usability.

For collection of forgery samples 25 volunteers were invited to be the attackers in the second part of the experiment. They were asked to attack the 20 users in the first part by providing their imitating finger-drawn PIN samples under two attacks in the threat model: PIN attack and Imitation attack. Note that five attackers were users from the first part, but they were made sure to not imitate themselves.

For the PIN attack, a page containing printed PINs of 20 users was given to each attacker. Then each attacker was asked to draw each PIN five times. Since attackers did not know how the original user drew, they drew each PIN in their own style. On an average, each user had 116 attack samples and, in total, 2330 samples for the PIN Attack were collected.

For the Imitation attack, attackers were presented with an animation of how the PIN was drawn by a user. Attackers were told to learn from this animation the shape and position of each digit, the drawing speed and acceleration, and anything that was helpful for imitation. They then trained themselves as long as they wanted on a Nexus phone with the DRAW-A-PIN system. Then they were asked to imitate the user five times. Attackers could also look at the animation and came back to the training phase anytime during imitation. On average, 116 imitating samples for each user and, in total, 2325 samples for the Imitation Attack were collected.

6.2. *Performance of the PIN content analyzer*

As described in Section 5, the Digit Recognizer is customized to each user by creating his or her own digit training set. The performance of the Digit Recognizer was evaluated using all the login digits from that user. Table 2 shows the average recognition rate of individual digits from 0 to 9 from all users' PIN samples in the data set with a k-NN classifier where k was set to different values. As it can be seen from the table, the system achieves the best recognition performance at $k = 7$ with an average accuracy of 99.57% and recognition rates of all digits were above 99%. $k = 7$ was set for the rest of the evaluation.

For PIN Content Analyzer (PCA), a PIN is accepted if its digits are recognized correctly and in correct order. Overall, the PCA achieves an average accuracy of 98.51% on all users which means it would reject a user two times in every 100 trials.

Table 2 – Recognition rate (%) of Digit Recognizer on each digit and recognition rate (%) of PCA at different *k* (number nearest-neighbor templates). The system achieves best recognition rate at *k* = 7.

Digit	0	1	2	3	4	5	6	7	8	9	PCA
k = 1	98.60	97.55	99.17	95.18	96.73	93.97	99.30	99.27	99.54	98.46	92.47
k = 3	99.25	98.66	99.17	96.05	98.31	96.04	99.30	99.69	99.26	98.94	94.68
k = 5	99.57	99.68	99.31	98.14	98.87	97.36	99.15	99.76	98.86	98.94	96.15
k = 7	99.57	99.84	99.45	99.34	99.77	99.53	99.38	99.76	99.83	99.27	98.51
k = 9	99.57	99.76	99.45	97.92	98.99	97.36	98.84	99.33	99.14	98.05	95.41

6.3. *Performance of the drawing behaviour analyzer*

As per normal practice in a biometric-based system, verification performance of Draw-A-PIN is reported in terms of a Receiver Operating Characteristic (ROC) curve, True Positive Rates (TPR), False Positive Rates (FPR), False Negative Rates (FNR) and Equal Error Rate (EER) (Sae-Bae et al., 2012, 2014; Shahzad et al., 2013; Sherman et al., 2014; Sun et al., 2014). An ROC curve captures the performance of a binary classifier system as its threshold is varied. It is derived by plotting TPR with respect to FPR at different threshold values. FNR is the percentage of legitimate login samples that are incorrectly rejected by the system. In contrast, FPR is the percentage of impostors' samples falsely accepted. In biometric systems, there is a known trade-off between security (FPR) and usability (FNR). That is, the more accurately the system rejects impostors, the more frequently the users being falsely rejected. EER is the error rate at which FPR and FNR are equal. The lower EER is, the better performance the system achieves.

For each user, his or her login PINs were used as positive samples, and imposter attempts from attackers were used as negative samples. The threshold for dissimilarity score is varied to compute FPR and FNR at different threshold values (operating point). Note that, unlike the pilot study, users have different numbers of positive samples depending on how often and how long they log into the system. Thus to prevent a user who has more number of positive samples to dominate the FNR (as well as TPR since TPR = 1- FNR) of the system, in this study FNR is computed on a per user-basis but based on the same threshold. Similar reasoning and calculation applied to FPR since users also have different numbers of negative samples (because five attackers were users and were made sure they did not imitate themselves). FPR, FNR of the system at each operating point are averages of these figures from all users. Here, EER is calculated based on these average values. Although EER was calculated with slight modification, it is in line with the general concept that corresponding points in ROC are derived at the same operating points for every user (Sae-Bae et al., 2012).

Table 3 depicts EERs of DBA under two attacks when different feature sets were used for the DTW calculation. It can be seen that DBA achieved the best performance with a full feature set of {X, Y, P, S} and worse performance with a feature set of only {X, Y} in both attacks. Thus, {X, Y, P, S} was used as the feature set for all evaluations from this point forward.

Table 3 – EER(%) of Draw-A-PIN under two attacks when using different feature sets.

	{X, Y}	{X, Y, P}	{X, Y, S}	{X, Y, P, S}
PIN attack	5.03	4.40	4.98	4.35
Imitation attack	19.97	16.26	17.52	14.96

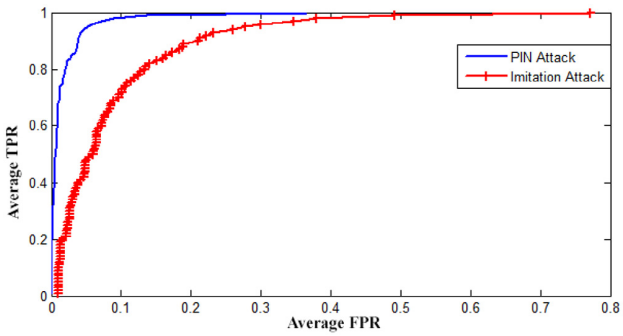

Fig. 3 – ROC curves of Draw-A-PIN in two attacks.

ROC curves of Draw-A-PIN under the two attacks are plotted in Fig. 3. The EER of Draw-A-PIN under PIN attack is 4.35% which means that the system rejects the attackers 95.65% even though they know the PINs. This is encouraging since compared with regular PIN-based system in the same scenario, the attack success rate is 100%. Under Imitation Attack, EER is 14.96%. Recall that in the Imitation Attack, the attacker had all the time he wanted to train himself before imitating the users and he could look at the animation anytime during the imitation. The increase in success rate is not surprising as the attackers have tremendous knowledge about drawing behavior of the user in addition to the PIN. Yet the system still rejected the attackers 85%. These results show that Draw-A-PIN could mitigate the risk of shoulder surfing.

6.4. *Performance of draw-a-PIN using invisible finger-drawn PIN*

When a user draws an invisible PIN, it is hard for the attacker to observe the PIN, but it may also affect authentication performance in the sense that the user might not remember which digits she has drawn and which digits she needs to draw next, since there is no feedback on the screen as a reminder. This was observed in some invisible PIN samples where users drew the same digit twice and the PINs were rejected. Moreover, with the visual feedback, the user can correct her digit by clearing the screen and redraw if she does not satisfy with that digit. As described in data collection section, a set of 599 Invisible PIN samples from users were collected. Performance of Draw-A-PIN using this set as positive samples, and the same set of imitation samples from two attacks in the previous section was evaluated. As shown in Table 4, Draw-A-PIN achieves slightly

Table 4 – The performance of DRAW-A-PIN using *Visible* and *Invisible* finger-drawn PIN.

Drawing mode	EER (%)	
	PIN attack	Imitation attack
Visible PIN	4.48	14.85
Invisible PIN	5.08	15.01

better performances with Visible PINs. The EER increases 0.6% under PIN Attack and 0.26% under Imitation Attack using Invisible PINs. Although Invisible PIN is an interesting idea, current results are not enough to conclude that it has better shoulder surfing resilience than that of Visible PIN. Hence, we leave an exploration of this for future work.

6.5. *Effectiveness of template updating strategy*

The term "template aging" refers to the situation in which the error rate of a biometric-based system increases over time between the acquisition of the enrollment sample and acquisition of the testing sample. This problem was investigated in DRAW-A-PIN and the result was reported in Fig. 4. The EER increases from 2.48% to 6.93% from the first day to the 10th day of the experiment under PIN Attack, and from 8.28% to 16.21% under Imitation Attack. Recall that the experiment was designed to simulate realistic scenarios where users draw PINs in different environments, this may also contribute to the variation of drawing behavior.

One possible solution to this problem is to employ a template updating strategy. One strategy is to update the training set with new samples that were accepted as the user logged in over time. This strategy was evaluated with the data set as follows. For each user, his or her login samples were divided into multiple sets of 5 consecutive login samples. Then the EER of each set of user training samples was calculated using the remaining samples from the same user as the genuine test samples and imitating samples were from the two attacks described earlier.

Fig. 5 shows the trend of EER of DRAW-A-PIN in this experiment. In both attacks, EER decreases over time. This result implies that PIN verification performance of users improved over time once user's drawing behavior became more consistent. This also suggests that updating drawing templates could help improve system performance.

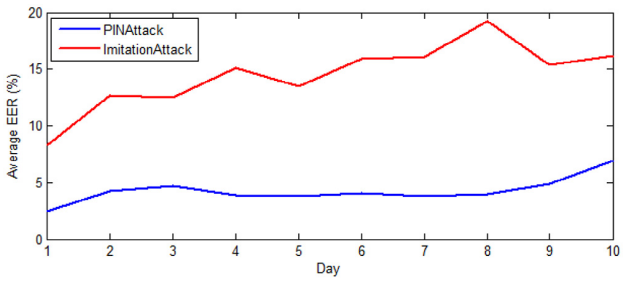

Fig. 4 – The EER of the system over 10 days of the experiment.

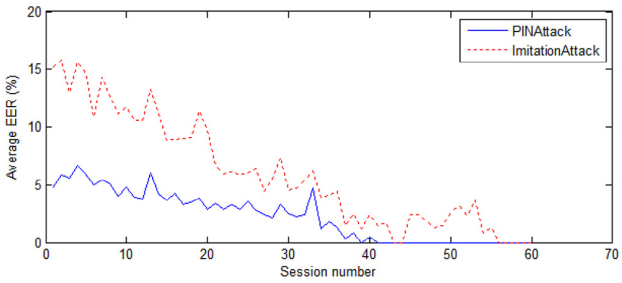

Fig. 5 – The EER of the system over different sessions when using template updating strategy. In each session, five login samples were used for training, and all login samples after training set and imitating samples were used for testing.

6.6. *Overall performance of DRAW-A-PIN*

Performance of individual component (PCA and DBA) in DRAW-A-PIN system was detailed in previous subsections. In this subsection, overall performance of the system is reported. Recall that a finger-drawn PIN is accepted if and only if it passes both PCA and DBA verification as described in Section 3.

Table 5 details overall performance of DRAW-A-PIN in term of EER under two attacks. It can be seen that these results are similar to the performance of DBA. This is because PCA has a very high accuracy (98.51%), therefore, the performance of DRAW-A-PIN mostly depends on the performance of DBA.

6.7. *Verification time*

Verification time includes the time it takes to draw the PIN and the running time of PCA and DBA components. Since the running times of DTW and $P recognizer are small – less than 1 s (approximately 500 ms in total, implemented and tested on a Nexus 5 phone), verification time mostly indicates the time it takes a user to draw his or her PIN.

Fig. 6 shows the distribution of drawing time of digits by 20 users in the experiment. The average value is 745 ms with a standard deviation of 263 ms. This means that users usually draw a digit in less than 1 s. Fig. 7 shows the distribution of drawing time of PINs drawn by 20 users. The average time a user takes to draw a 4-digit PIN is 6.12 s with a standard deviation of 1.49 s. This indicates that there were some gaps in drawing between each digit of the PIN. The user could take a small break between each digit. Some PINs were drawn in about 11 s. One possible reason for this is that the user may draw wrong digits and then he or she had to redraw them. Other reason may be the user was interrupted in the middle of the drawing because he or she was also involving with other

Table 5 – Overall performance of DRAW-A-PIN when combining PIN Content Analyzer and Drawing Behaviour Analyzer.

	EER
PIN attack	4.84%
Imitation attack	14.11%

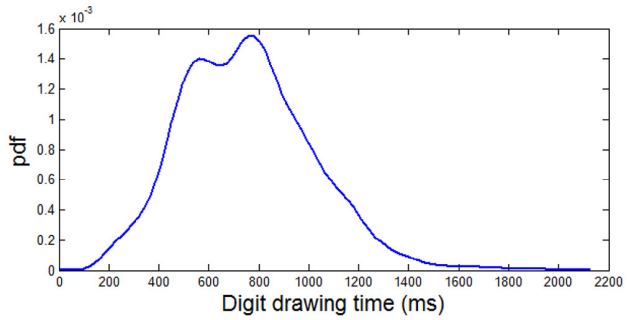

Fig. 6 – The distribution of drawing time of digits by 20 users.

activities such as talking with someone. Compared to regular PIN authentication using 4-digit PIN, Harbach et al. reported an average authentication time of 4.7 s ($sd = 20.72s$, $median = 2.85s$) in 2014 (Harbach et al., 2014) and recently reported an average of 2.67 s ($sd = 1.14s$, $median = 2.41s$) including time to recover from typing errors in 2016 (Harbach et al., 2016) with a larger dataset. It is obvious that drawing PINs takes longer time. To understand the impact of this delay to system usability, users were asked on their perception of the login time on Draw-A-PIN and most of them said that they did not feel the system is slow. Interestingly, some said that they did not pay attention to the time because they just enjoyed drawing their PINs. This effect also was observed in related work (Yadav et al., 2015) and explained in psychological science (Gable and Poole, 2012; Noulhiane et al., 2007) in which studies have found that pleasant experiences seem to shorten perceptions of time.

6.8. *Comparison with existing schemes*

The performance of Draw-A-PIN was compared with well-known schemes in the same direction reported in De Luca et al. (2012), Sae-Bae and Memon (2014), and Shahzad et al. (2013) under a similar attack where the attacker already knows the secret or gesture of the user. Shahzad et al. (2013) proposed GEAT that verifies users based on a set of 10 most effective gestures using machine learning approach. Using per-user EER calculation, they reported an average EER of 4.02% for GEAT, which is the average of EERs of all users. Adopting the same calculation, Draw-A-PIN achieved an average EER of 3.46% which is slightly lower.

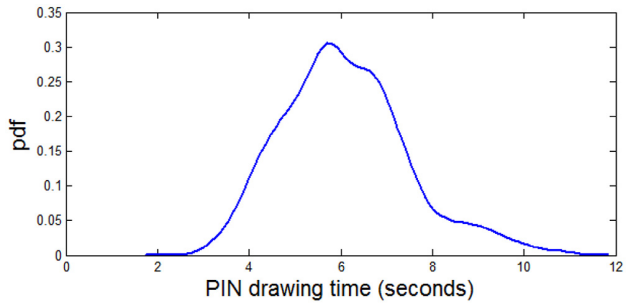

Fig. 7 – The distribution of drawing time of 4-digit PINs by 20 users.

De Luca et al. (2012) improved Android locking pattern by adding an extra layer of verification based on a set of features including pressure, size, and speed of drawing of a user. They also used DTW as verification algorithm. Compared to their overall accuracy of 90.8%, which is the average of best accuracy figures calculated by per-user basis for five top users with mostly zero false positives (as reported in fig. 5, page 8 in De Luca et al. (2012)), Draw-A-PIN achieves a much better accuracy of 96.71%. Note that, for comparison purposes the overall accuracy of Draw-A-PIN was calculated using the method employed by De Luca et al. (2012) presented below. Also, finger-drawn PINs of each user used for this calculation included visible and invisible samples.

$$Accuracy = \frac{\sum TP + \sum TN}{\sum TP + \sum TN + \sum FP + \sum FN} \quad (8)$$

where TP is True Positives, FP is False Positives, TN is True Negatives, FN is False Negatives. Technically, Luca et al. calculated this figure at the operating point where TPR was maximized for each user, whereas here it is calculated at the point of EER of each user.

Sae-Bae and Memon (2014) proposed to verify a user based on a signature drawn on a touch screen. Their reported EER of 4.37% is slightly lower than that of Draw-A-PIN (4.84%). The two figures were calculated in the EER calculation method as presented in Section 6.

These results show that Draw-A-PIN has comparable performance with existing schemes while the system could have more chance of real-world adoption as it can leverage user's familiarity with PINs verification system.

6.9. *Usability evaluation*

System Usability Scale (SUS) tool (Brooke, 1996) was used to gather subjective assessments about the usability of Draw-A-PIN system. Hundreds of studies have used SUS as a standard usability metric (Bangor et al., 2008, 2009). Moreover, Tullis et al. found that SUS gives the most reliable usability metric results (Tullis and Stetson, 2004). SUS includes 10 questions in which the response to each question is given on a five-point scale ranging from "strongly disagree" to "strongly agree". An overall SUS score is a value in the range of 0–100, where a higher value is more desired because it indicates a more usable system. The raw SUS score can be transformed into a percentile (Sauro, 2011) and an A-F grading scale (Bangor et al., 2008) for better interpretation of the result. The ten SUS questions are as follows.

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.

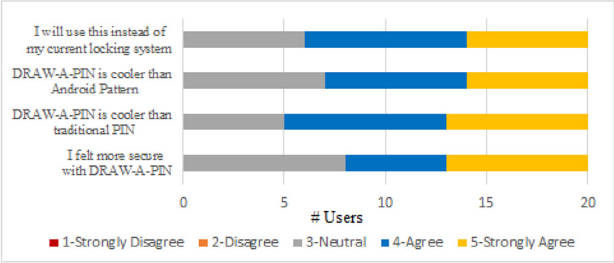

Fig. 8 – Users' rating on the security, "cool factor" and likelihood of future usage of Draw-A-PIN.

9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

Four Likert scale questions were also added to ask about perceived security, "cool factor" in comparison with PIN and Android locking patterns, and likelihood of future usage. In particular, four questions are:

11. I felt more secure using this system to lock my devices.
12. I felt this system is cooler than the traditional PIN locking system.
13. I felt this system is cooler than the Android locking pattern system.
14. If this system becomes a product, I will use it instead of my current locking system.

Note that these four additional questions are not included when calculating SUS score.

Overall, Draw-A-PIN achieves a SUS score of 82 which is much higher than the standard average score of 68 (Sauro, 2011), and could be converted to a percentile rank of 90% with a SUS letter grade of A. Furthermore, Fig. 8 shows that users rated Draw-A-PIN as secure, cooler and may be preferable over conventional PIN and Android locking methods. Users also showed that they will probably use the system in the future. These results confirm the usability of Draw-A-PIN.

7. Limitations and discussion

One limitation of this work is the relatively small number of subjects used in the experiments. But it should be noted that our experiment was designed to collect realistic data. Often, data are collected in a lab environment where participants are asked to provide genuine samples within a short time frame and in a supervised and consistent setting (Shahzad et al., 2013; Sherman et al., 2014; Sun et al., 2014). In contrast, our samples were collected from everyday use of the system in an unsupervised setting to better reflect the performance that might be achieved in a real deployment. Getting a large number of users to complete such a study is difficult.

Draw-A-PIN was evaluated under stronger threat models than previous studies. Specifically, one of our attacks was to accurately animate how the PIN is being drawn by a genuine user on the screen. This is a stronger attack model than just showing the video of hand movement as previously done; since, in the latter case, the information used to verify the digit would likely be occluded by the user's hand.

Also, there is still a lot of room to improve the proposed approach. First, as shown in Section 6.5, the template set can be updated gradually to increase verification performance as the system adapts to a user's drawing behavior variation over time. An appropriate update strategy needs to be investigated. Second, the verification algorithm can perhaps be improved by exploring more sophisticated techniques that may require additional computation. These topics are dedicated for future work.

In terms of usability, it would be interesting to investigate the relative merits of a full enrollment versus a shortened re-enrollment process should users wish to change their PIN. For example, should the re-enrollment only ask the user to draw digits that were not present in previous PIN(s)? This idea is also left for future investigation.

Lastly, even though there are many related proposals and studies on gesture-based authentication, we argue that drawing a PIN is an attractive option as compared to drawing other gestures or patterns. The main reason for this is that users are familiar with PIN and 59% still use a PIN for authentication (based on our survey presented in Section 3). Additionally, PIN will continue to be used in many systems such as ATM and debit-credit card transactions and users will continue to maintain familiarity with them. Another interesting example is that PIN is still used to protect the encryption key on iOS devices despite the fact that physiological biometric verification is offered to users (Apple Inc., 2015). Thus, we believe that Draw-A-PIN is a reasonable choice on touch-based devices in real-world applications and could be offered as an option to entering the PIN in some situations.

8. Conclusion

This paper presents Draw-A-PIN, a one-step two-factor user authentication system for touchscreen devices and demonstrates the potential of exploiting distinctive traits in drawing digits for user authentication. The performance of the system was evaluated on a data set consisting of everyday use of Draw-A-PIN, from 20 volunteers over 10 days and imitation samples from 25 attackers. The results show that Draw-A-PIN achieved an EER of 4.84% even when the attacker already knew the PIN of a user. This indicates that Draw-A-PIN can potentially address attacks where the user PIN is known. In the scenario of a targeted attack where attackers have an animation of an exact reproduction of how the PIN was drawn by the user and could train themselves before imitation, the proposed system still rejected attackers at a rate of 85%. These results show that Draw-A-PIN provides resilience against shoulder surfing attacks. In terms of usability, the System Usability Scale test on our users suggested that Draw-A-PIN is usable and may be preferable over conventional PIN and locking pattern methods. Draw-A-PIN, however, still has limitations on digit recognition rate and its performance in behavioral biometric verification can be improved. Therefore, future work can include the improvement of digit recognizer and finding a better algorithm for utilizing additional characteristics from drawing behavior in order to provide better recognition performance.

Acknowledgment

This material is based in part on work supported by the National Science Foundation under Grant No. 1228842 and by Thailand Research Fund under Grant No. 5980078. We would like to thank the reviewers for their constructive feedback.

REFERENCES

Apple Inc. iOS Security, iOS 9.0 or later; 2015. Available from: https://www.apple.com/business/docs/iOS_Security_Guide.pdf. [Accessed 12 August 2016].

Arora SS, Cao K, Jain AK, Paulter NG Jr. Design and fabrication of 3d fingerprint targets, Information forensics and security, IEEE transactions on, 2016.

Aviv AJ, Gibson K, Mossop E, Blaze M, Smith JM. Smudge attacks on smartphone touch screens, in Proceedings of the 4th USENIX conference on offensive technologies, pp. 1–7, USENIX Association, 2010.

Bailador G, Sanchez-Avila C, Guerra-Casanova J, de Santos Sierra A. Analysis of pattern recognition techniques for in-air signature biometrics. Pattern Recognit 2011;44(10):2468–78.

Balzarotti D, Cova M, Vigna G. Clearshot: eavesdropping on keyboard input from video, in Security and privacy, 2008. SP 2008. IEEE symposium on, pp. 170–183, 2008.

Bangor A, Kortum PT, Miller JT. An empirical evaluation of the system usability scale. Int J Hum Comput Interact 2008;24(6):574–94.

Bangor A, Kortum P, Miller J. Determining what individual sus scores mean: adding an adjective rating scale. J Usability Stud 2009;4(3):114–23.

Biggio B, Akhtar Z, Fumera G, Marcialis GL, Roli F. Security evaluation of biometric authentication systems under real spoofing attacks. IET Biom 2012;1(1):11–24.

Bonneau J, Preibusch S, Anderson R. A birthday present every eleven wallets? the security of customer-chosen banking pins, in Financial cryptography and data security, vol. 7397, pp. 25–40, Springer Berlin Heidelberg, 2012.

Bonneau J, Herley C, Oorschot PCV, Stajano F. The quest to replace passwords: a framework for comparative evaluation of web authentication schemes, in Proceedings of the 2012 IEEE symposium on security and privacy, SP '12 (Washington, DC, USA), pp. 553–567, IEEE Computer Society, 2012.

Brooke J. Sus-a quick and dirty usability scale. Usability Eval Ind 1996;189:194.

Cao K, Jain AK. Hacking mobile phones using 2d printed fingerprints, 2016.

Cardoso A, Wichert A. Handwritten digit recognition using biologically inspired features. Neurocomputing 2013;99:575–80.

Chaos Computer Club. Chaos computer club breaks apple touchid; 2013. Available from: https://www.ccc.de/en/updates/2013/ccc-breaks-apple-touchid. [Accessed 12 August 2016].

Ciresan DC, Meier U, Gambardella LM, Schmidhuber J. Deep big multilayer perceptrons for digit recognition, in Neural networks: tricks of the trade, pp. 581–598, Springer, 2012.

Connell SD, Jain AK. Template-based online character recognition. Pattern Recognit 2001;34(1):1–14.

De Luca A, Hang A, Brudy F, Lindner C, Hussmann H. Touch me once and I know it's you!: implicit authentication based on touch screen patterns, in Proceedings of the SIGCHI conference on human factors in computing systems, pp. 987–996, ACM, 2012.

Dunphy P, Yan J. Do background images improve draw a secret graphical passwords? in Proceedings of the 14th ACM conference on computer and communications security, pp. 36–47, ACM, 2007.

Fatindez-Zanuy M. On the vulnerability of biometric security systems, IEEE aerospace and electronic systems magazine (2004), pp. 3–8, 2004.

Feinberg A. The evolution of ATM skimmers; 2014. Available from: http://gizmodo.com/the-terrifying-evolution-of-atm-skimmers-1626794130. [Accessed 12 August 2016].

Feng T, Zhao X, Carbunar B, Shi W. Continuous mobile authentication using virtual key typing biometrics, in Trust, security and privacy in computing and communications (TrustCom), 2013 12th IEEE international conference on, pp. 1547–1552, 2013.

Frank M, Biedert R, Ma ED, Martinovic I, Song D. Touchalytics: on the applicability of touchscreen input as a behavioral biometric for continuous authentication. IEEE Trans Inf Forensics Sec 2013;8:136–48.

Gable PA, Poole BD. Time flies when you're having approach-motivated fun effects of motivational intensity on time perception. Psychol Sci 2012;23(8):879–86.

Geuss M. Police can demand a suspect unlock a phone with a fingerprint; 2014. Available from: http://arstechnica.com/tech-policy/2014/10/virginia-judge-police-can-demand-a-suspect-unlock-a-phone-with-a-fingerprint/. [Accessed 12 August 2016].

Gupta P, Behera S, Vatsa M, Singh R. On iris spoofing using print attack, in Pattern recognition (ICPR), 2014 22nd international conference on, pp. 1681–1686, 2014.

Harbach M, von Zezschwitz E, Fichtner A, De Luca A, Smith M. Its a hard lock life: a field study of smartphone (un) locking behavior and risk perception, in Symposium on usable privacy and security (SOUPS), 2014.

Harbach M, De Luca A, Egelman S. The anatomy of smartphone unlocking: a field study of android lock screens, in Proceedings of the 2016 CHI conference on human factors in computing systems, pp. 4806–4817, ACM, 2016.

Jermyn I, Mayer AJ, Monrose F, Reiter MK, Rubin AD. The design and analysis of graphical passwords, in Usenix security, 1999.

Kim J, Sin B-K. Online handwriting recognition, in Handbook of Document Image Processing and Recognition, pp. 887–915, Springer, 2014.

Kravets D. Forcing suspects to reveal phone passwords is unconstitutional, court says; 2015. Available from: http://arstechnica.com/tech-policy/2015/09/forcing-suspects-to-reveal-phone-passwords-is-unconstitutional-court-says/. [Accessed 12 August 2016].

LeCun Y, Jackel L, Bottou L, Brunot A, Cortes C, Denker J, et al. Comparison of learning algorithms for handwritten digit recognition, in International conference on artificial neural networks, vol. 60, 1995.

Liu C-L, Nakashima K, Sako H, Fujisawa H. Handwritten digit recognition: benchmarking of state-of-the-art techniques. Pattern Recognit 2003;36(10):2271–85.

Liu J, Zhong L, Wickramasuriya J, Vasudevan V. User evaluation of lightweight user authentication with a single tri-axis accelerometer, in Proceedings of the 11th international conference on human-computer interaction with mobile devices and services, p. 15, ACM, 2009a.

Liu J, Zhong L, Wickramasuriya J, Vasudevan V. uwave: accelerometer-based personalized gesture recognition and its applications. Pervasive Mob Comput 2009b;5(6):657–75.

Maggi F, Volpatto A, Gasparini S, Boracchi G, Zanero S. A fast eavesdropping attack against touchscreens, in Information assurance and security (IAS), 2011 7th international conference on, pp. 320–325, 2011.

Maiorana E, Campisi P, González-Carballo N, Neri A. Keystroke dynamics authentication for mobile phones, in Proceedings of

the 2011 ACM symposium on applied computing, pp. 21–26, ACM, 2011.
Matsumoto T, Matsumoto H, Yamada K, Hoshino S. Impact of artificial 'gummy' fingers on fingerprint systems, in Proceedings of SPIE, vol. 4677, 2002.
Mendaza-Ormaza A, Miguel-Hurtado O, Blanco-Gonzalo R, Diez-Jimeno F. Analysis of handwritten signature performances using mobile devices, in Security technology (ICCST), 2011 IEEE international Carnahan conference on, pp. 1–6, 2011.
Muslukhov I, Boshmaf Y, Kuo C, Lester J, Beznosov K. Know your enemy: the risk of unauthorized access in smartphones by insiders, in Proceedings of the 15th international conference on human-computer interaction with mobile devices and services, MobileHCI '13 (New York, NY, USA), pp. 271–280, ACM, 2013.
Nguyen TV, Sae-Bae N, Memon N. Finger-drawn pin authentication on touch devices, in Image processing (ICIP), 2014 IEEE international conference on, pp. 5002–5006, 2014.
Noulhiane M, Mella N, Samson S, Ragot R, Pouthas V. How emotional auditory stimuli modulate time perception. Emotion 2007;7(4):697.
Prabhakar S, Pankanti S, Jain AK. Biometric recognition: security and privacy concerns. IEEE Secur Priv 2003;2:33–42.
Riley S. It's me, and here's my proof: why identity and authentication must remain distinct; 2006. Available from: http://technet.microsoft.com/en-us/library/cc512578.aspx. [Accessed 12 August 2016].
Sae-Bae N, Memon N. Online signature verification on mobile devices. IEEE Trans Inf Forensics Sec 2014;9:933–47.
Sae-Bae N, Memon N. Quality of online signature templates, in Identity, security and behavior analysis (ISBA), 2015 IEEE international conference on, pp. 1–8, IEEE, 2015.
Sae-Bae N, Ahmed K, Isbister K, Memon N. Biometric-rich gestures: a novel approach to authentication on multi-touch devices, in Proceedings of the SIGCHI conference on human factors in computing systems, pp. 977–986, ACM, 2012.
Sae-Bae N, Memon N, Isbister K, Ahmed K. Multitouch gesture-based authentication. IEEE Trans Inf Forensics Sec 2014;9(4):568–82.
Sakoe H, Chiba S. Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans Acoust 1978;26:43–9.
Sauro J. Measuring usability with the system usability scale (sus); 2011. Available from: http://www.measuringu.com/sus.php. [Accessed 12 August 2016].
Schuckers SAC. Spoofing and anti-spoofing measures. Inf Sec Tech Rep 2002;7:56–62.
Security Research Labs, Spoofing Fingerprints. Spoofing fingerprints; 2013. Available from: https://srlabs.de/spoofing-fingerprints/. [Accessed 12 August 2016].
Serwadda A, Phoha VV. When kids' toys breach mobile phone security, in Proceedings of the 2013 ACM SIGSAC conference on computer & communications security, pp. 599–610, ACM, 2013.
Shahzad M, Liu AX, Samuel A. Secure unlocking of mobile touch screen devices by simple gestures: you can see it but you can not do it, in Proceedings of the 19th annual international conference on mobile computing & networking, pp. 39–50, ACM, 2013.
Sherman M, Clark G, Yang Y, Sugrim S, Modig A, Lindqvist J, et al. User-generated free-form gestures for authentication: security and memorability, in Proceedings of the 12th annual international conference on mobile systems, applications, and services, MobiSys '14 (New York, NY, USA), pp. 176–189, ACM, 2014.
Shukla D, Kumar R, Serwadda A, Phoha VV. Beware, your hands reveal your secrets, Proceedings of the 2014 ACM SIGSAC conference on computer & communications security (CCS), 2014.
Sun J, Zhang R, Zhang J, Zhang Y. Touchin: Sightless two-factor authentication on multi-touch mobile devices, arXiv preprint arXiv:1402.1216, 2014.
Tian J, Qu C, Xu W, Wang S. Kinwrite: handwriting-based authentication using kinect, in NDSS, 2013.
Tullis TS, Stetson JN. A comparison of questionnaires for assessing website usability, in Usability professional association conference, pp. 1–12, 2004.
Vatavu R-D, Anthony L, Wobbrock JO. Gestures as point clouds: a $p recognizer for user interface prototypes, in Proceedings of the 14th ACM international conference on multimodal interaction, pp. 273–280, ACM, 2012.
Whitney L. Samsung galaxy s5 fingerprint scanner thwarted by hack; 2014. Available from: https://www.cnet.com/news/samsung-galaxy-5-fingerprint-scanner-thwarted-by-hack/. [Accessed 12 August 2016].
Yadav DK, Ionascu B, Ongole SVK, Roy A, Memon N. Design and analysis of shoulder surfing resistant pin based authentication mechanisms on Google glass, in International conference on financial cryptography and data security, pp. 281–297, Springer, 2015.
Yue Q, Ling Z, Fu X, Liu B, Yu W, Zhao W. My Google glass sees your passwords!, Black Hat USA, 2014.

Toan Nguyen is a PhD Candidate at the Department of Computer Science and Engineering at NYU School of Engineering. Before joining NYU, he was a Lecturer at School of Information and Communication Technology (SoICT), Hanoi University of Science and Technology (HUST). His research interests include cybersecurity, usable security, computer networking, machine learning, human–computer interaction, and biometrics. He earned a Bachelor of Engineering in Information Technology and a Master of Science in Computer and Communication Engineering from HUST in 2009 and 2011, respectively. He is a Fellow of the Vietnam Education Foundation Fellowship Program.

Napa Sae-Bae is a lecturer in Department of Computer Science at Rajamangala University of Technology Suvarnabhumi. She received Bachelor of Engineer in Telecommunication Engineering department and Master of Engineer in Information Technology department from King Mongkut's Institute of Technology Ladkrabang, Thailand, and received the M.Sc. and PhD in Computer Science from New York University. Her research interests lie in the area of biometric, authentication, signal processing, pattern recognition, and consumer security. She has published 15 articles in journals and conference proceedings in different venues and holds a patent on multi-touch gesture for user-authentication on touch interface.

Dr. Nasir Memon is a professor in the Department of Computer Science and Engineering at NYU School of Engineering, director of the OSIRIS laboratory, and a founding members of the Center for Interdisciplinary Studies in Security and Privacy (CRISSP), a collaborative multidisciplinary initiative of several schools within NYU. His research interests include digital forensics, biometrics, data compression, network security and usable security. Dr. Memon received a Master of Science in Computer Science and a PhD in Computer Science from the University of Nebraska. He has published more than 250 articles in journals and conference proceedings and holds a dozen patents in image compression and security. He has won several awards including the Jacobs Excellence in Education award and several best paper awards. He has been on the editorial boards of several journals and was the Editor-In-Chief of Transactions on Information Security and Forensics. He is an IEEE Fellow and a distinguished lecturer of the IEEE Signal Processing Society. Memon is the co-founder of Digital Assembly and Vivic Networks, two early-stage startups in NYU's business incubators.