Towards Continuous and Passive Authentication Across Mobile Devices: An Empirical Study

Xiao Wang
Carnegie Mellon University
sean.wang@sv.cmu.edu

Tong Yu
Carnegie Mellon University
tong.yu@sv.cmu.edu

Ole Mengshoel
Carnegie Mellon University
ole.mengshoel@sv.cmu.edu

Patrick Tague
Carnegie Mellon University
patrick.tague@sv.cmu.edu

ABSTRACT

Mobile devices, such as smartphones and tablets, have become prevalent given their ample functionality brought by a variety of applications. Unfortunately, these devices face security and privacy threats due to unauthorized access. Ordinary protection mechanisms such as passcode and fingerprint verification are widely employed to mitigate the threats. To achieve strong security without sacrificing usability, extensive research efforts have been devoted to continuous authentication through passive sensing and behavior modeling. Nowadays, more and more users own multiple devices. This trend presents opportunities for further optimization of authentication across devices. In this paper, we conduct an empirical study on how a behavioral model created on one device can be transferred to other devices to bootstrap continuous authentication. To pursue this goal, we collect 160 sets of usage data on multiple mobile devices and perform a proof-of-concept experiment. The results demonstrate that we can leverage the similarity between user behaviors on different devices to enable cross-device authentication and anomaly detection.

ACM Reference format:
Xiao Wang, Tong Yu, Ole Mengshoel, and Patrick Tague. 2017. Towards Continuous and Passive Authentication Across Mobile Devices: An Empirical Study. In *Proceedings of WiSec '17 , Boston, MA, USA, July 2017,* 11 pages.
DOI: 10.1145/3098243.3098244

1 INTRODUCTION

Mobile devices are now infused into various aspects of our daily lives including work, social activity and entertainment. A wide variety of smartphone and tablet apps enrich the personal lives of millions of users in a variety of ways. As a consequence of the extensive usage, ample private data is stored on or is accessible through mobile devices.

To protect such private data against threats imposed by unauthorized access, mobile devices and their operating systems employ traditional access control mechanisms using passcode, lock pattern and fingerprint. Previous work has shown drawbacks of these methods with regards to security, usability and cost [2][3][4][26]. To tackle the challenges, researchers propose *continuous and passive authentication* as a supplement to existing schemes [9][24][32]. The proposed system constantly senses a user's interactions with the device and authenticates the user in runtime through behavioral data. The behaviors typically include touchscreen gestures, frequent physical locations, poises holding or picking up the device and so forth. They can collectively contribute to all-round protection against unauthorized users. Embedded sensors on devices, such as touchscreen, accelerometer and gyroscope, facilitate the collection of behavior-related measurements.

Though extensive efforts have been devoted into continuous authentication, the rapid and massive development in the mobile industry brings new opportunities and corresponding challenges. Many customers now own multiple devices. According to market research, about 31% of US adults have both smartphones and tablets [1], while UK households typically own on average three types of internet-enabled devices [20]. Mobile users switch between their devices in daily usage. It raises the problem of user authentication *across* multiple mobile devices in a *seamless* manner.

In support of the goal of cross-device user authentication, we study in this paper the possibility of *transferring a behavioral model learned on a user's device to her other devices to bootstrap the authentication process.* Suppose, for example, that a user has been using one mobile device for a long period of time, and the device has collected sufficient measurements and built a model. When this user purchases a new mobile device, either as a supplement or a replacement for the first device, the second device needs to accumulate behavioral measurements for its owner and construct a new model, leaving the device a hiatus without the protection of continuous authentication and incurring extra overhead of building user profiles from scratch. To optimize for seamless cross-device authentication, it is desirable to transfer such models across devices without needing to collect additional data to re-train the model on the new device, allowing us to leverage the prior knowledge attained from the first device. This capability can help bootstrap trust on the new device. Aside from the specific application scenario of cross-device authentication, this paper reveals the underlying connections of behavioral patterns on different devices.

To build behavioral models, we leverage users' touchscreen gestures. On most mobile devices, the touchscreen serves as the main

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
WiSec '17 , Boston, MA, USA
© 2017 ACM. 978-1-4503-5084-6/17/07…$15.00
DOI: 10.1145/3098243.3098244

interface between users and devices, offering opportunities to collect rich behavior data and to identify useful patterns. Typically, users have to use the touchscreen during normal usage of the device. Hence, using touchscreen data for continuous authentication incurs minimal cost, while sensing user activities constantly with accelerometer or gyroscope adds significant runtime cost since the data is not necessary for the device's normal functionality. Our work studies how to extract useful behavioral patterns that persist across multiple devices, including data preprocessing to handle inherent measurement nuances due to hardware differences and feature design for classification algorithms.

Contributions. As a first attempt to study continuous authentication in the multi-device scenario, we have conducted a proof-of-concept experiment. Our experiment leverages a real-world app for data collection so that we can approximate usage patterns in normal usage of devices. To understand the cross-device behavioral patterns, each participant in our experiment uses the experimental app on eight devices of four different models. Our results show that it is promising to reuse existing model on an old device to bootstrap continuous authentication on a new device.

We summarize our contributions as below:

- We make the first attempt to study continuous and passive authentication across mobile devices in contrast to earlier work on authentication on individual devices.
- We design effective behavioral features to capture similarity of user behaviors across multiple devices.
- We collect real-world behavior data and evaluate our approach to demonstrate how a behavior model learned on one device can be transferred to other devices.

Roadmap. The paper is organized as follows. We contrast our work with previous literature in Section 2. In Section 3, we present our system and adversary models, and an experimental app for touchscreen data collection. Section 4 presents the cross-device continuous authentication framework and discusses the feature extraction and classification methods. We show our experimental setup and evaluation results in Section 5. We conclude in Section 6.

2 RELATED WORK

Mobile devices are now equipped with various sensors including touchscreen, accelerometer, gyroscope and so forth. With this sensing capability, mobile devices are aware of environmental factors and user actions, and they can respond accordingly. This awareness also provides the opportunity of authenticating users through sensing. The basic idea is to leverage sensory data to model user behaviors and distinguish the current user from the expected user.

Through passive sensing, a mobile device can obtain user biometrics for authentication. Biometrics can be categorized as either physiological or behavioral [15]. Physiological biometrics include physical human attributes such as fingerprint and iris, while behavioral biometrics include aspects of how people behave, including gait and keystroke. Behavioral biometrics have the advantage of requiring only a simple collection of inexpensive sensors commonly included in modern mobile devices [21]. In contrast, physiological biometrics like fingerprint and iris serve as stronger identifiers. However, these explicit identifiers raise privacy concerns and are not welcomed in many circumstances. Moreover, they require extra user effort and may thus interfere with user experience.

The line of research starts from user authentication on single devices through different kinds of sensory measurements. Feng et al. [8] introduced a touchscreen-based approach that authenticates users through finger gestures and achieved satisfactory performance. Meng et al. [17] [16] focused on touch dynamics and experimented on Nexus One with their touch biometrics. Similarly, Vu et al. [27] proposed a user identification and authentication approach through exploiting capacitive touchscreen. Frank et al. [9] and Xu et al. [29] investigated continuous and passive authentication based on the way a user interacts with the touchscreen. Jain et al. [11] developed an authentication mechanism by analyzing a user's behavioral traits modeled by acceleration data, curvature of swipes and finger area. Later, Li et al. [13] designed a mechanism to re-authenticate current users based on finger movements on the screen. Draffin et al. [6] modeled users' micro-behavior when they are typing on the soft keyboard on the screen, including touch location on each key, drift from finger down to finger up and touch pressure. Sae-Bae et al. [23] showed that multi-touch gestures are applicable to user authentication. De et al. [4] collected data from user input on a touchscreen, and used dynamic time warping on the data for identification. Dey et al. [5] leveraged accelerometer data to extract frequency domain features to obtain unique fingerprints. Zhu et al. [32] proposed a system that uses accelerometers, gyroscopes and magnetometers to profile users. Our paper investigates user authentication across multiple devices, which differs from authentication on a single device. The new scenario introduces another dimension (device). Distinct device models of different sizes and hardware configurations introduce new challenges that render certain biometrics for the single-device case ineffective.

Cross-device patterns received academic attention in the online search domain. Wang et al. [28] examined cross-device task continuation from PC to smartphone for a particular sets of search tasks. They used topics and transition time to predict whether a search task is continuation from PC to mobile. Montañez et al. [18] studied search behaviors and device transition features to predict cross-device search transitions. Karlson et al. [12] analyzed the usage log of desktops and mobile phones to understand patterns how people transition between devices. Our work concentrates on cross-device behaviors on mobile apps. We believe there are still many research questions regarding how users and information interact with and transition between multiple devices. The research is also related to activity recognition [10][14][25] in the sense that both leverage sensory data to model user behaviors. Some new development in activity recognition using deep learning can potentially be helpful in behavior-based authentication [31][30].

3 BACKGROUND

In this section, we introduce the system and adversary models and technical assumptions. Also, we present the experimental app used in this study for touchscreen data collection.

3.1 System and Adversary Models

The mobile devices we consider are equipped with a touchscreen, through which a device can measure a user's clicks and swipes on

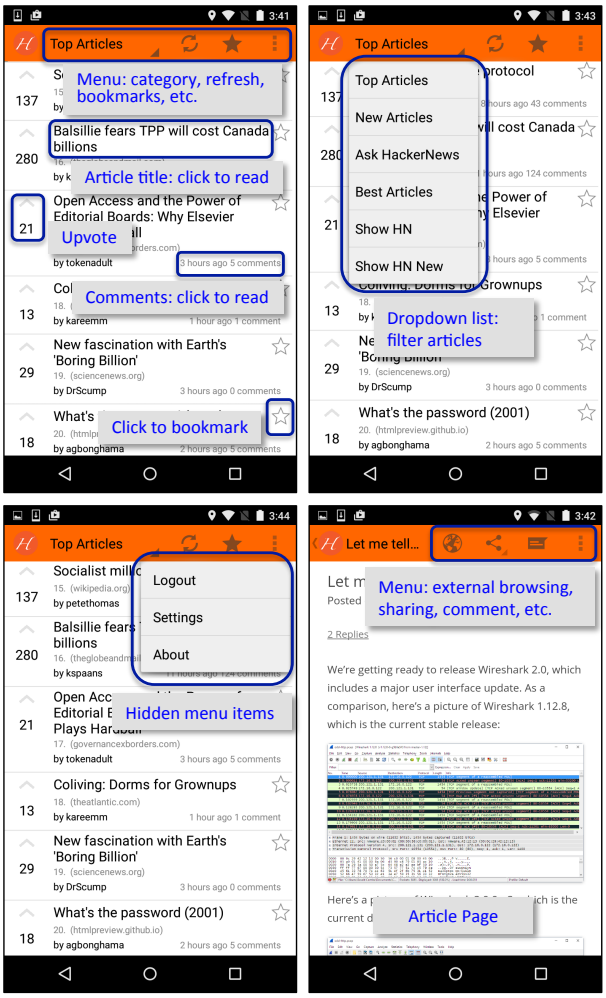


Figure 1: We show screenshots of our experimental app, which is a popular news reader. The app manifests mainstream design components of mobile apps such as list layout and menu bar. Its functionalities enable ample interactions between the device and its users. Our system collects user clicks and swipes on screen while using the app.

the screen. The timestamped gesture data can hence be used to model user behaviors while interacting with the device.

We consider a common adversary model in which an attacker has gained access to a user's mobile device. The device is either unprotected (e.g., without screen locked) or the adversary has obtained the authentication secret such as passcode or fingerprint through, for example, social engineering [29]. Thereafter, the attacker can access apps and their data on the device, violating the user's privacy.

3.2 The Experimental App

Previous research studies continuous authentication on single devices through touchscreen analytics. To collect measurements from users, they employ their own experimental app with particularly designed user interfaces. For example, in *touchalytics* [9], subjects are instructed to use a simple image comparison app. They have to move the screen content through some touchscreen gestures in order to navigate between images. In another work by Xu et al. [29], the authors designed an app that instructs participants to perform pre-defined operations such as typing a sentence using the soft-keyboard and writing down a character on the screen.

In this paper, we instead *make use of a real-world app to avoid constraints in controlled lab settings.* Moreover, the touchscreen gestures consequently incorporate contextual information of the design and content of the app. That means we can analyze the gestures with respect to the app context instead of as isolated gestures. For instance, a user's click locations may exhibit certain patterns due to the layout of the user interface. This, however, does not mean the detection is only performed when this specific app is being used. It serves as an example of how the operating system on the device can model user gestures with regards to different contexts. Moreover, some common gestures such as swipes are app-independent and can be applied to other apps. Further, many apps share similar layout as we will discuss shortly. The app we experiment with is an open-sourced Hacker News Android app [22]. The app is a popular client on mobile devices for browsing YC Hacker news [19]. We show the layout and functions of the app in Figure 1. For content display and delivery, the app renders a list of news articles for users to scroll and browse. One reason we choose to experiment with this app lays in the fact that it provides adequate functions such as article filtering by category, upvoting/downvoting, sharing and bookmarking, with which the app enables ample interactions between the app and its users, creating opportunities of finding effective patterns to distinguish the owner from unauthorized users.

This app manifests typical design principles of mobile apps on the market. For instance, it employs the list-plus-menu-bar layout that is especially suitable for content display and page transition on small-screen devices. Hence, although our approach is presented based on this app, the underlying techniques can be generalized to many other apps. This means the system can keep collecting touchscreen measurements when the attacker is using different apps on the device or simply swiping on the home screen, providing continuous authentication. In our model, we leverage a user's click and swipe patterns as features since they are two fundamental gestures to interact with apps on a mobile device.

To obtain experimental data, we instrument the app so that it records user interactions with the app. Additionally, we set up a server to collect usage-related measurements. The app stores those measurements locally on the device before uploading them to our server. Our instrumented code only runs when the app is open, which reduces runtime overhead and prevents draining battery. Moreover, the data file is uploaded only through WiFi networks to avoid cellular data transfer. The instrumented code collects timestamped measurements of user actions while using the app, including clicks and swipes.

3.3 Touchscreen Data Format

In this paper, to model a user's usage pattern, we leverage click and swipe data. When using our experimental app, users have to swipe on the screen to scroll the list of news articles in order to

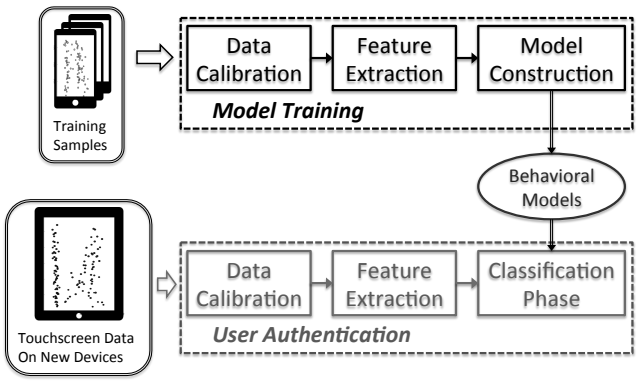


Figure 2: We present an overview of the cross-device authentication process, in which the owner behavioral model is trained on old devices and subsequently applied to bootstrap continuous authentication on new devices.

find an article of interest. Then, they click on the article title to proceed to the article page. To support normal functions of a mobile device, when we touch the screen of a device, dedicated hardware automatically generates data record and reports it to the operating system as raw events. The system or the app further process the raw data to understand user gestures and respond accordingly. Taking Android as an example, each raw event contains measurements of position, size, pressure and timestamp of a screen touch. Position is measured in terms of pixels along two dimensions of the device. It is fine-grained, while size and pressure information is less accurate. Some devices do not have a dedicated sensor for pressure measurements. Therefore, they either report a constant value or simulate a value from the size measurement.

In our data files, clicks and swipes are recorded using the following format

`timestamp,x_pos,y_pos,pressure,size`.

A click is one single tap on the screen, while a swipe consists of a sequence of taps. We can segment a tap series into individual clicks or swipes based on inter-click intervals.

To obtain ground truth, we modify the app so that each user is prompted for login credentials the first time the user opens the app on a device. A user is required to register username and password which are later used to log into the app on other devices. We can therefore obtain labeled data for our experiment.

4 BEHAVIOR-BASED CROSS-DEVICE USER AUTHENTICATION

In this section, we present our cross-device authentication mechanism as shown in Figure 2. With thoughtful design of data manipulation and feature engineering, the approach aims to extract useful behavioral patterns that persist across multiple devices.

4.1 Measurement Calibration

Hardware differences introduce inherent dissimilarity of measurements across devices. Instead of leaving it fully to our learning model to automatically compensate for the difference, we calibrate some difference beforehand using heuristics and domain knowledge. To give an example, for the pressure and size data, sensors on different devices have distinct sensitivity and measurement reference. With the tap data of the same user on different devices, we can learn the relative difference across devices and normalize the data in terms of mean and variance. For the position of clicks, we create relative values in addition to the deterministic values to facilitate generation of features that focus on relative positions. The calibration is generally helpful even for devices of the same model since sensory measurements might not be aligned across them because of hardware differences.

4.2 Feature Engineering

Before training a model, we handcrafted useful features. Among all sensory measurements, we rely on user clicks and swipes in this paper. They are two primary gestures of users to interact with devices. In Figure 3, we visualize the clicks and swipes of three users on two devices.

In our app, there are icons for users to click on. Since icon positions are fixed, their corresponding click measurements have negligible variance on position, and are thus less favorable as features. For articles, users scroll the list to find articles of interest, and click on the titles. In this sense, click positions have more variance and entropy for different users. As shown in Figure 3(a), users have different click positions. User 1's clicks spread the screen. User 2 might be left-handed due to the imbalance to the left side of the screen. User 3 appears to use left and right hand equally. Users swipe on the screen to browse the article list. A swipe is essentially a sequence of taps. We also fit a circle to each swipe to show its radius and direction. As shown in Figure 3(b), swipes can be another indicator of handedness; user 2 is most likely left-handed while users 1 and 3 are the opposite. Despite that, user 3's swipes have smaller radius and are more concentrated compared to user 1, which adds further biometric information.

To obtain discriminating information from user clicks and swipes, we create 120 features in both time domain and frequency domain. The features are computed from a group of clicks and swipes. We list some of the features in Table 1.

To preliminarily assess individual features, we use each single feature to classify users and show the *area under the curve* (*AUC*) value in Figure 4. The *AUC* value is between 0 and 1, and can indicate the effectiveness of a feature in binary classification (the higher the better). We can observe that some features are relatively useful with *AUC* above 0.9. For the click actions, the most important features include the mean value and variance of click trace in vertical and horizontal directions. Meanwhile, for the swipe actions, touch size and pressure are also very important, as well as the mean value and variance of swipe trace in vertical and horizontal directions.

4.3 Machine Learning Models

To detect unauthorized users in multi-device scenarios based on behavioral patterns, we apply supervised learning techniques. In this section, we describe the two machine learning models applied in this paper: *support vector machines* and *random forest*.

4.3.1 The Support Vector Machines. SVM is a binary classification model. It aims to find an optimal hyperplane that best

ID	Feature Description
fc_1, fc_2	The mean of click position in vertical and horizontal directions.
fc_3	The mean of click pressure.
fc_4	The mean of click size.
fc_5, fc_6	The variance of click position in vertical and horizontal directions.
fc_7, fc_8	Ratio of vertical clicks or horizontal clicks to all clicks.
$fc_9, \cdots, fc_{28}$	In horizontal or vertical directions, the index (frequency) of the 10 highest FFT value of click data.
$fc_{29}, fc_{33}, fc_{37}$	The max, min and median values of click position in vertical direction.
$fc_{30}, fc_{34}, fc_{38}$	The max, min and median values of click position in horizontal direction.
$fc_{31}, fc_{35}, fc_{39}$	The max, min and median values of click pressure.
$fc_{32}, fc_{36}, fc_{40}$	The max, min and median values of click size.
fs_1, fs_2	The mean of swipe trace in vertical and horizontal directions.
fs_3	The mean of pressure in swipes.
fs_4	The mean of touch area of taps in swipes.
fs_5, fs_6	The variance of swipe trace in vertical and horizontal directions.
fs_7, fs_8	Ratio of vertical swipe or horizontal swipe actions among all swipe actions.
$fs_9, \cdots, fs_{28}$	In horizontal or vertical directions, the index (frequency) of the 10 highest FFT value of swipe data.
$fs_{29}, fs_{33}, fs_{37}$	The max, min and median values of swipe trace in vertical direction
$fs_{30}, fs_{34}, fs_{38}$	The max, min and median values of swipe trace in horizontal direction
$fs_{31}, fs_{35}, fs_{39}$	The max, min and median values of swipe pressure
$fs_{32}, fs_{36}, fs_{40}$	The max, min and median values of swipe touch area
fs_{41}	The angle of moving during swiping
$fs_{42}, fs_{46}, fs_{50}, fs_{54}$	The mean, max, min and median values of velocity during swiping in vertical direction
$fs_{43}, fs_{47}, fs_{51}, fs_{55}$	The mean, max, min and median values of velocity during swiping in horizontal direction
$fs_{44}, fs_{48}, fs_{52}, fs_{56}$	The mean, max, min and median speeds of swipe pressure change
$fs_{45}, fs_{49}, fs_{53}, fs_{57}$	The mean, max, min and median speeds of swipe touch area change
fs_{58}	The acceleration of moving during swiping in vertical direction
fs_{59}	The acceleration of moving during swiping in horizontal direction

Table 1: We summarize our features engineered from user clicks and swipes. The features include common statistics and frequency domain components. They capture characteristics of users interacting with the experimental app, and fulfill transfer of behavioral models across mobile devices.

separates data from two classes. Given two-class training data $(y_1, x_1), \cdots (y_l, x_l)$, where $y_i = \pm 1$, $\forall i$ are class labels, $x_i \in R^n$, $\forall i$ are training feature vectors, it optimizes the following weighted sum of the regularization term and training losses:

$$\min_{\mathbf{w}, b} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l} \max(1 - y_i(\mathbf{w}^T\phi(x_i) + b), 0),$$

where C is the penalty parameter, $\mathbf{w}$ is the model weight and b is the bias term. $\phi(x_i)$ is a projection function which can map the feature vector x_i to a higher dimensional space, to improve the class separability.

The decision value is defined by $f(x) = \mathbf{w}^T\phi(x)+b$. In prediction, given a test data with feature vector x_k, if $f(x_k) > 0$, we predict it as +1, otherwise we predict it as −1.

4.3.2 The Random Forest. Another type of machine learning model we study is random forest. It combines the results from multiple decision trees. Depending on whether the label is categorical or continuous value, it can address either classification or regression problems.

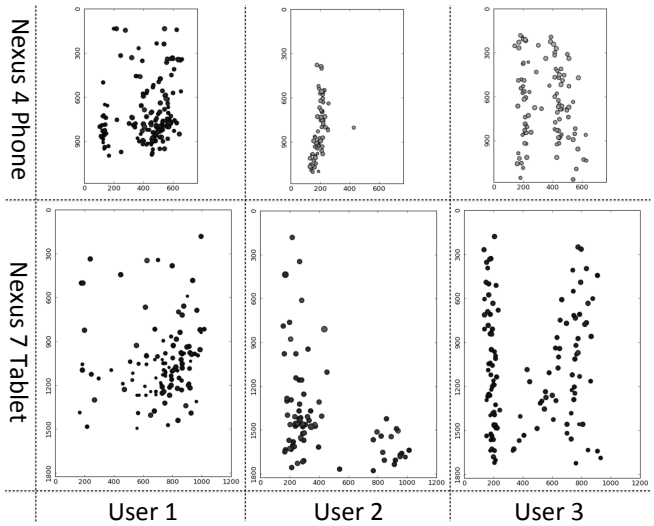
In decision tree, the feature space is split by a tree structure. Every time when we split the space by some dimension, we choose the dimension leading to highest information gain after splitting. Appropriate reducing depth of tree, which is referred as Tree Pruning, can effectively avoid overfitting in practice.

The random forest model consists of multiple decision trees. In classification problem, mode of the classes by individual trees is output. In a regression problem, mean prediction by individual trees is the prediction. This ensemble technique can also effectively help avoid overfitting, which can be caused by the extreme depth of some individual decision tree models.

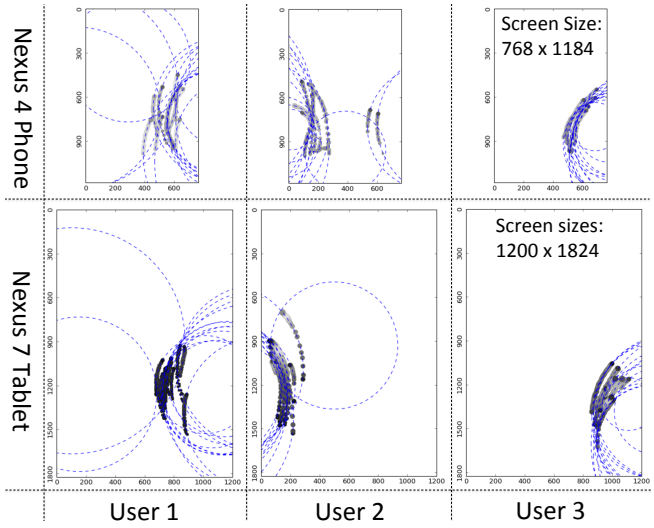
When the data is imbalanced, we can provide each data sample with a weight associated with classes to the random forest model. In detail, if the ith training sample belongs to class j, the weight w_i for the ith training sample is

$$\frac{l}{n_classes \times l_j}$$

where l is the amount of training data, l_j is the amount of training data in class j, and $n_classes$ is the number of classes in this problem. With this weighting method, the random forest will give higher weights to those samples appearing less frequently in the training dataset.



(a) A tap has four parameters including horizontal and vertical positions on screen, size and pressure. Each point in above scatter plots represents a tap. The size of a point visualizes the size of the corresponding tap. The pressure is visualized as darkness.



(b) A swipe is a sequence of taps. In the figure, for each user on each device, we randomly select 15 swipes for illustration. We fit a circle to each swipe to show its radius and direction.

Figure 3: To provide some visual insights, we depict users' taps on screen when clicking on article titles, and swipes when scrolling the article list, which are randomly and uniformly sampled from each user's tap and swipe data.

5 EXPERIMENTAL RESULTS

In this section, we present our experimental results to demonstrate the feasibility of cross-device anomaly detection. We aim to show how a device can distinguish its owner from other users such as family, friends and strangers, by leveraging a classification model obtained by another device of the owner. We start with a description of our data collection and experimental settings including 1) how we train a model on one single device and 2) how we apply the model on other devices to detect unauthorized users. We assess our approach by widely-used criteria such as AUC and F_1 scores. We report the cross-device authentication performance, and also contrast it to the performance of anomaly detection on single devices. Moreover, we provide analytic results regarding several practical considerations in applying the approach to real practice.

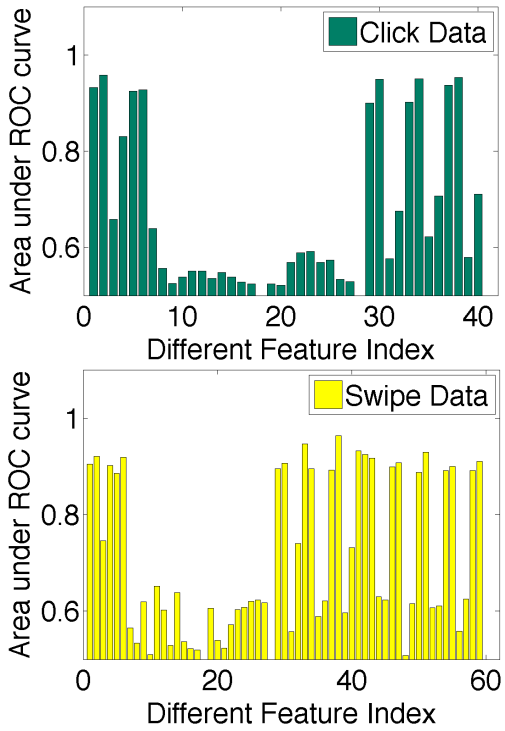


Figure 4: To briefly evaluate the effectiveness of our features, we use each single feature separately to classify users and compare the AUC of each feature on the click data and swipe data, respectively. The x-axis gives the feature index while the y-axis shows the corresponding AUC. Some features outperform the others, achieving the AUC of over 0.9. The various features can collectively enable cross-device authentication of mobile users.

5.1 Data Collection

In this cross-device authentication study, each subject uses the experimental app on four models of devices: Nexus S, Nexus 4, Nexus 7-2012 and Nexus 7-2013. For each model, a subject is required to use the app on two devices of that model. In total, we obtain usage data of each subject on eight devices. Though it is rare for a real user to own eight devices, with this setup we do not actually mean for them to have eight devices. Four pairs of different devices allow us to study usage behaviors of each subject on 1) identical devices, 2) same type but different models (Nexus S phone *vs.* Nexus 4 phone), and 3) different types of devices (Nexus 4 phone *vs.* Nexus 7 tablet).

In addition to behavioral patterns of each subject across eight devices, we also need to study the variation of behaviors on the

ID	Model	Type	Screen Size
P1a	Nexus S	Phone	400 × 800
P1b	Nexus S	Phone	400 × 800
P2a	Nexus 4	Phone	768 × 1184
P2b	Nexus 4	Phone	768 × 1184
T1a	Nexus 7 (2012)	Tablet	800 × 1205
T1b	Nexus 7 (2012)	Tablet	800 × 1205
T2a	Nexus 7 (2013)	Tablet	1200 × 1824
T2b	Nexus 7 (2013)	Tablet	1200 × 1824

Table 2: We list all eight devices in our experiment which will be referred to by their ID in later sections. The devices are of different types and screen sizes that are believed to affect user behaviors.

same devices by different subjects. To this end, we recruit twenty subjects from different demographic groups to participate in the data collection, giving essentially 160 sets of usage data in terms of the $\langle user, device \rangle$ pair. With the data, we are able to study inter-user, intra-user, inter-device and intra-device behavioral similarity or difference. That said, we admittedly note that it is favorable to include more devices and subjects into this user study. However, our effort is limited by the constraints of time and hardware. We do not have dozens of mobile devices for simultaneous data collection from many subjects, and collecting data on eight devices for even one user is already time-demanding. In this paper, we hope to deliver proof-of-concept results on cross-device authentication, which we believe can be fulfilled with the 160 sets of usage data.

After we distribute devices to a subject for data collection, we do not give specific instructions to use the experimental app other than a basic introduction of the app. They use the app in accordance to their own preference and habit. In this fashion, we hope the subjects exhibit their own patterns in the usage that are not biased by any particular recipe. For the twenty subjects, we have collected hundreds to thousands data points of clicks and swipes.

Last, due to limited resources, the device models we use are only a small portion of all device models. The four device types in our experiment consist of phones and tablets of different models, sizes and hardware. Although they are not completely representative of all devices, they present differences we hope to have and deliver meaningful results.

5.2 Settings and Evaluation Criteria

For one device, we choose $N_s = 1$ subject in our dataset as the owner and treat the other subjects as unauthorized users. In our experiment, we have all twenty users use the app on each device so that we have negative instances for each device during the training phase. In most real world circumstances, although only the owner uses her device, the app can still easily obtain negative training instances for that device by using the data of other users on their own devices of the same model.

Our experiments focus on two scenarios: single-device and cross-device authentication. For the single-device case, the training and test data is from the same devices. We split the data from each device, and use 75% of the data as training samples and the rest as test samples. In the cross-device scenario, the training and test data is from different devices. We defer details to the corresponding sections about the two scenarios. Additionally, we also evaluate the effect of N_s on the authentication performance as it is possible that several users share one device. In this case, there are more than one owner. The positive training instances are from multiple users.

The performance of authentication is evaluated by metrics including area under the curve (*AUC*) and F_1 score. The *AUC* is defined as the area of the region under the *ROC* curve, where the *ROC* curve is used to show the performance of a binary classifier against its classification threshold. The x-axis of the *ROC* curve is the false positive rate and the y-axis is the true positive rate. The F_1 score is defined as:

$$F_1 = \frac{2TP}{2TP + FP + FN},$$

where *TP* is the true positive, *FP* denotes the false positive, *FN* is the false negative.

In anomaly detection, if an unauthorized user is classified as owner, it is considered as a false positive. Meanwhile, if the true owner is classified as owner, it is regarded as a true positive. Classifiers can be tuned to accommodate different requirements for the true positive, false positive, true negative and false negative. In addition to these common evaluation metrics, there are some more statistically robust metrics such as the Gini coefficient [7] that can capture different error distributions and hence provide more insights for performance evaluation and feature selection.

5.3 Unauthorized User Detection on Single Devices

We start with a basic scenario, where the device attempts to detect unauthorized users on each single device. Though it is not the primary focus of this paper, the results can be contrasted with the cross-device scenario for readers to better understand how the proposed approach performs for cross-device authentication. We first give an example to explain our setup for the experiment. To evaluate the authentication performance on device P1a, we choose user 1 as its owner, and other subjects as unauthorized users. To train the classification model, we use 75% of user 1's data as positive instances and 75% of other users' data as negative instances. Model evaluation is conducted on the remaining 25% of data. Then we iteratively choose each user as the owner and repeat the training and test process. The final result for each device is an average value over all twenty users.

We show the *AUC* values of the random forest model on all eight devices in Figure 5. We can observe that it achieves *AUC* scores from 0.80 to 0.96 with both click and swipe features. The click features offer more discriminating power, while on some devices the swipe features are more effective. Previous research shows that swipe pattern is a better indicator of user identity than click pattern. However, in our experiment, clicks can be equally important, and even outperforms swipes in some cases. This is because we employ a real-world app so that user clicks have more contextual information given the context of the app. For example, users scroll a list of articles and then click on an article of interest. The click position is not fixed due to the context of the app. This variation provides entropy for distinguishing different users. In this sense, a click is

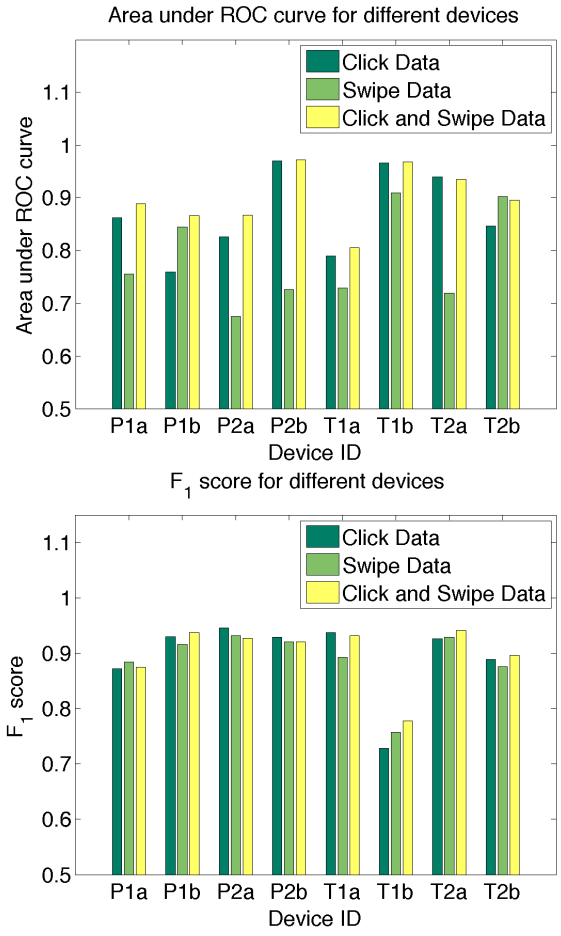


Figure 5: We plot the *AUC* and F_1 score of unauthorized user detection on each of the eight devices. We also contrast the performance when only click features, only swipe features, and both types of features are used, respectively.

no longer merely a tap on the screen. We can create more features based on user clicks with respect to app context. In terms of F_1 score, for most devices, our approach achieves higher than 0.87 value. Click and swipe features contribute similar performance. A fusion of the click and swipe features enhances the performance on some devices, while the improvement is marginal on the others. This might be caused by the poor quality of one type of features. If some features cannot well represent the usage behaviors or the behavioral patterns are inconsistent in terms of these features, the incorporation of them can bring no benefit or even compromise the discriminating power of the model.

Furthermore, we evaluate the performance of different machine learning models with three feature settings: only click features, only swipe features, and both click and swipe features.

Table 3 shows the *AUC* and F_1 scores of different machine learning models. We find that random forest provides the best performance in this classification task. Especially, it outperforms other machine learning models significantly in the case where only click

	Models	Click	Swipe	Both
AUC	Kernel SVM	0.7923	0.8094	0.8785
	Decision Tree	0.6974	0.5392	0.6650
	Random Forest (10 trees)	0.8711	0.7864	0.8742
	Random Forest (50 trees)	0.8711	0.7864	0.9012
	Random Forest (100 trees)	0.8711	0.7864	0.9013
F_1	Kernel SVM	0.9543	0.9538	0.9541
	Decision Tree	0.4878	0.4853	0.4864
	Random Forest (10 trees)	0.6483	0.6822	0.6607
	Random Forest (50 trees)	0.8608	0.8545	0.8666
	Random Forest (100 trees)	0.8979	0.8889	0.9045
	Random Forest (400 trees)	0.9382	0.9439	0.9440

Table 3: We compare the *AUC* and F_1 scores of different machine learning models with different features. Random forest provides the best performance among all classifiers considered in our experiment.

data is available. In the case where only swipe data is available, random forest gives similar performance as kernel SVM model. We can also observe that as the number of trees increases in the random forest model, we obtain improvements in the *AUC* value. For the sake of space, in the following experiments, we will use the random forest model as our classifier.

5.4 Unauthorized User Detection Across Multiple Devices

In this section, we present our results for the multi-device scenario. One device leverages the user behavior model obtained on another device to perform authentication. The training phase is similar to that in the single device scenario. The learned model is thereafter applied to a new device and tested using sensory data collected on that new device.

Considering the diversity of those devices, we present the performance for each device in Figure 6. The results are averaged across all twenty users. The model obtained on one device is applied to the other seven devices. We can observe that in the multi-device scenario, the detection performance is sensitive to the pair of mobile devices used for training and test. If the two mobile devices are of the same model, we can typically expect high *AUC* values since user behaviors are likely to be consistent across them. For example, if the model is trained on device P1a, we can observe a higher *AUC* score on the test data collected from device P1b than that on the data from the other devices. Similar observations can be found between device P2a and P2b. The behavior similarity can be explained by the fact that devices of the same model share identical size and the same touchscreen sensors. As a result, the training data and test data is more likely to follow the same statistical distribution. This property facilitates behavior modeling and improves authentication performance.

However, this observation is not strictly true. For example, our approach achieves lower performance for device pair (T1a, T1b) and (T2a, T2b) compared to device pair (P1a, P1b) and (P2a, P2b). The reason might be that compared with small-screen mobile phones,

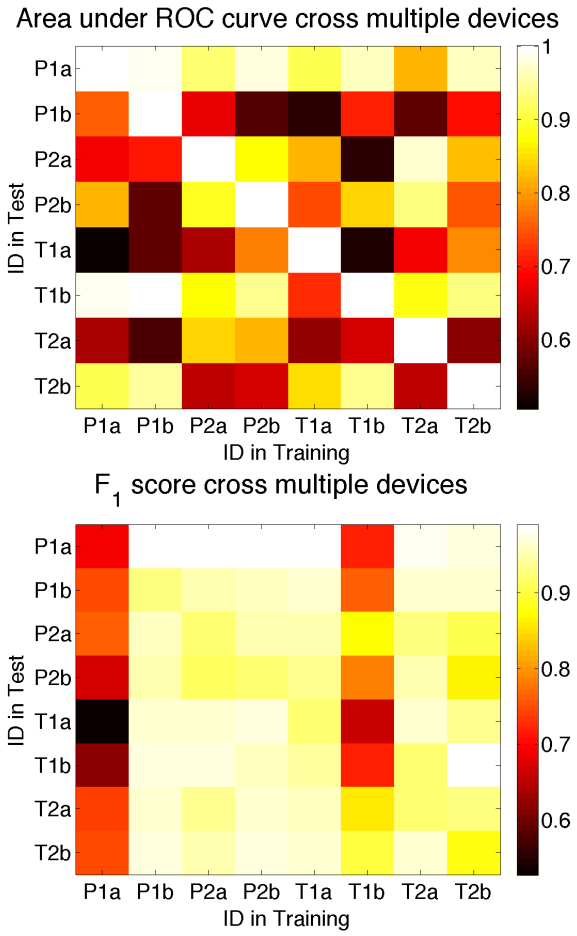


Figure 6: The AUC and F_1 score of authentication cross devices. Each row (column) represents a device. We report the AUC in (i, j) where training data is from device i and test data from device j.

tablets have larger screens which render user behaviors on the touchscreen relatively diverse and unpredictable.

At the end of this section, we summarize the average detection performance on single device and cross devices in Table 4 given different sets of features. In general, we find that $AUC_{\text{single}} > AUC_{\text{cross}}$. This inferior performance in the cross-device scenario can be explained by the differences in user behaviors across multiple devices and therefore the gap between the distributions of the training data and test data. Another observation is that in the cross devices scenario, the click features are still very useful for distinguishing users. Clicks bear contextual information with regard to the layout of the app (positions of clicks) and exhibit consistency across different devices. Swipe features underperform probably because swipe is a more complex gesture with more variations which may not maintain across devices. Moreover, screen size might have greater impact on the swipe gesture.

	Scenarios	Click	Swipe	Both
AUC	Single-Device	0.8725	0.7986	0.9135
	Cross-Device	0.8053	0.7023	0.8104
F_1	Single-Device	0.8987	0.9083	0.9132
	Cross-Device	0.8967	0.8958	0.9052

Table 4: For single-device and cross-device authentication, we compare the AUC and F_1 scores in the presence of different features.

5.5 Practical Considerations

In this section, we extend our discussion to cover some practical considerations for the mechanism to be applied to real practice. To this end, we investigate the following questions with the data we collected.

(1) How much click or swipe data is sufficient for our system to detect unauthorized users? (Section 5.5.1)
(2) To build an effective detector, how should we select our training data? (Section 5.5.2)
(3) What if a mobile device is owned by more than one user? (Section 5.5.3)

5.5.1 Required Clicks or Swipes. In applications of using the click or swipe data to detect unauthorized users, we would like to study the effects of test data size N on the performance. We prefer a small N for prompt detection. If someone is using an insensitive app, the device can wait a longer time for more behavioral data.

To detect unauthorized users in an online fashion, a classification decision is made after every click or swipe.

Models are obtained on one device and then applied to another. We report the AUC scores of three different scenarios: a) only click data is available, b) only swipe data is available, c) both click and swipe data is available. In the experiments, the app gradually receives more data and reports the major prediction on test data. We apply the data of all 19 unauthorized users to test this online detection scheme. We calculate the recall of unauthorized users, which is defined as the number of correctly detected unauthorized users divided by 19.

Figure 7 shows the results. We can observe that as the amount of test data increases, our model enjoys higher AUC values of unauthorized users. Similar as the observations in Section 5.4, the click data is more helpful than swipe data to identify unauthorized users on new devices where the labeled data is very limited. In the case when the test data amount is extremely small, the click data is more valuable to help identify unauthorized users.

5.5.2 The Amount of Training Data. In this section, we will discuss the effects of training data on the detection performance. Due to the diversity in devices, the engineered features across different devices have diverse distributions. An effective way to improve the performance is to include more data for training.

For example, when the system performs detection on a Nexus phone, we have two different strategies to train the model for classification.

- Strategy 1: We can train the model on the user's data from this Nexus phone. The training data's distribution may be

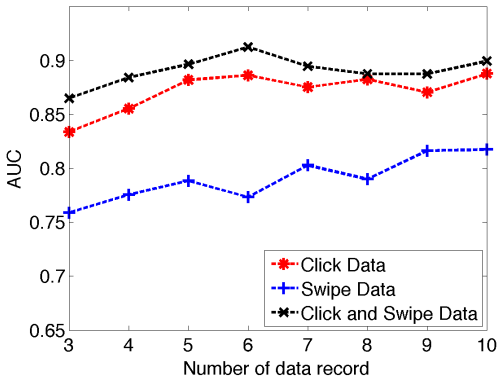


Figure 7: We show the *AUC* scores in online detection when the amount of validation data increases. The detection performance gets enhanced with more data observed. However, it flattens out gradually.

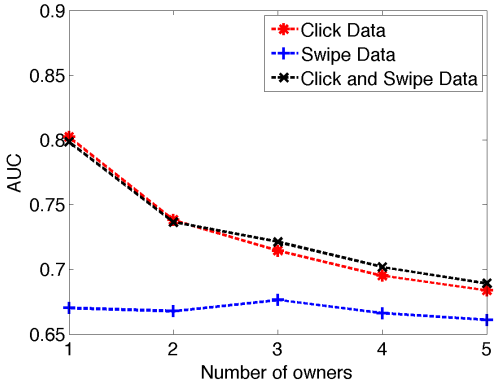


Figure 9: The *AUC* score decreases when the number of owners increases. The increased complexity of behavioral patterns introduced by different users can confuse the authentication algorithm.

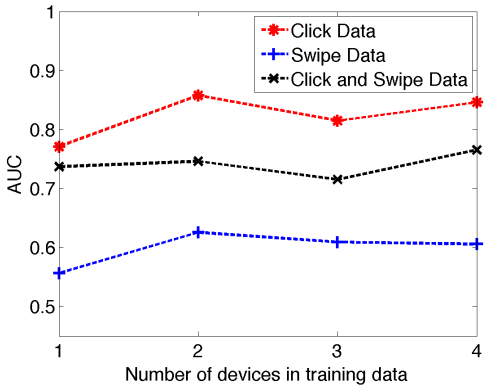


Figure 8: The *AUC* score changes as there is more training data. The outcome of adding training data from more devices can be complicated as user behaviors vary across devices.

very similar to the test data. However, labeled data for this particular user on this Nexus phone may be limited.

- Strategy 2: We can train the model on the user's data from all types of devices he used. The distribution of the training data is diverse yet might be slightly different from the test data. However, the labeled data for this particular user are fruitful.

In our experiments, each time we fix one device for test and use the data from other devices to train our model. We increase the number of devices in training data and report *AUC* on test data. We run this experiment for every device and show the average *AUC*. Figure 8 compares the results of the two different strategies. In x-axis we change the number of devices in the training data and in y-axis the *AUC* is reported. Generally, given a device for test, adding more data is helpful to improve the authentication performance. However, more data may not always help on some devices where users' click and swipe actions are different from those in other devices.

5.5.3 Multiple Owners For One Device. We also consider real world scenarios where mobile devices may have multiple owners. Consider an example of all members of a family sharing a tablet at home. It is worthy of investigation the case where positive data comes from more than one user on a single device.

Figure 9 depicts the *AUC* values against increasing number of owners. We observe that as the number of users increases, the *AUC* score decreases. The result indicates that the detection task becomes more difficult when we want to distinguish multiple owners on one single device. The reason is that when there are more owners, the distribution of positive data becomes more diverse, which can confuse the binary classifiers since the combined behaviors hide the patterns of individual users.

6 CONCLUSION AND FUTURE WORK

In this paper, we study the problem of translating models for user authentication across multiple dissimilar mobile devices, focusing in particular on app-independent features such as click and swipe behavior. Previous research focuses on identifying unauthorized users on individual devices, while we investigate whether a user authentication model obtained on one device can be applied to another device of that owner for the purpose of bootstrapping trust on new devices. We collect 160 sets of real usage data. We handcraft useful features from user clicks and swipes, which capture subtle user behavior patterns that are preserved across devices. In our experiment of detecting unauthorized users on different devices, we achieve an *AUC* score of 80% to 96%.

We envision several future directions to explore. First, we want to study how user behaviors might change over time and its subsequent impact on user authentication. Second, we are interested in extending our experiment to other mobile apps and devices. Third, it is valuable to investigate user behaviors across different operating systems such as Android *vs.* iOS. This new dimension might impose new challenges.

REFERENCES

[1] Tom Rosenstiel Amy Mitchell and Leah Christian. 2014. Mobile Devices and News Consumption: Some Good Signs for Journalism. (2014).

[2] Adam J Aviv, Katherine Gibson, Evan Mossop, Matt Blaze, and Jonathan M Smith. 2010. Smudge Attacks on Smartphone Touch Screens. *WOOT* 10 (2010), 1–7.

[3] Erika Chin, Adrienne Porter Felt, Vyas Sekar, and David Wagner. 2012. Measuring user confidence in smartphone security and privacy. In *the Proceedings of the Eighth Symposium on Usable Privacy and Security*. ACM, 1.

[4] Alexander De Luca, Alina Hang, Frederik Brudy, Christian Lindner, and Heinrich Hussmann. 2012. Touch me once and i know it's you!: implicit authentication based on touch screen patterns. In *the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 987–996.

[5] Sanorita Dey, Nirupam Roy, Wenyuan Xu, Romit Roy Choudhury, and Srihari Nelakuditi. 2014. Accelprint: Imperfections of accelerometers make smartphones trackable. In *the Proceedings of the Network and Distributed System Security Symposium (NDSS)*.

[6] Benjamin Draffin, Jiang Zhu, and Joy Zhang. 2014. Keysens: passive user authentication through micro-behavior modeling of soft keyboard interaction. In *the Mobile Computing, Applications, and Services*. Springer, 184–201.

[7] Simon Eberz, Kasper B. Rasmussen, Vincent Lenders, and Ivan Martinovic. 2017. Evaluating Behavioral Biometrics for Continuous Authentication: Challenges and Metrics. In *the Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (ASIA CCS '17)*. ACM, New York, NY, USA, 386–399. DOI: https://doi.org/10.1145/3052973.3053032

[8] Tao Feng, Ziyi Liu, Kyeong-An Kwon, Weidong Shi, Bogdan Carbunar, Yifei Jiang, and Ngac Ky Nguyen. 2012. Continuous mobile authentication using touchscreen gestures. In *the IEEE Conference on Technologies for Homeland Security (HST)*. IEEE, 451–456.

[9] Michael Frank, Ralf Biedert, En-Di Ma, Ivan Martinovic, and Dong Song. 2013. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *the IEEE Transactions on Information Forensics and Security* 8, 1 (2013), 136–148.

[10] Jay Prakash Gupta, Nishant Singh, Pushkar Dixit, Vijay Bhaskar Semwal, and Shiv Ram Dubey. 2013. Human activity recognition using gait pattern. *the International Journal of Computer Vision and Image Processing (IJCVIP)* 3, 3 (2013), 31–53.

[11] Ankita Jain and Vivek Kanhangad. 2015. Exploring orientation and accelerometer sensor data for personal authentication in smartphones using touchscreen gestures. *the Pattern Recognition Letters* (2015).

[12] Amy K Karlson, Brian R Meyers, Andy Jacobs, Paul Johns, and Shaun K Kane. 2009. Working overtime: Patterns of smartphone and PC usage in the day of an information worker. In *The Pervasive Computing*. Springer, 398–405.

[13] Lingjun Li, Xinxin Zhao, and Guoliang Xue. 2013. Unobservable Re-authentication for Smartphones.. In *the Network Distributed System Security*.

[14] Lin Liao, Dieter Fox, and Henry Kautz. 2006. Location-based activity recognition. *the Advances in Neural Information Processing Systems* 18 (2006), 787.

[15] Wenchao Meng, Duncan Wong, Steven Furnell, and Jianying Zhou. 2015. Surveying the Development of Biometric User Authentication on Mobile Phones. (2015).

[16] Yuxin Meng, Duncan S Wong, and others. 2014. Design of touch dynamics based user authentication with an adaptive mechanism on mobile phones. In *the Proceedings of the 29th Annual ACM Symposium on Applied Computing*. ACM, 1680–1687.

[17] Yuxin Meng, Duncan S Wong, Roman Schlegel, and others. 2013. Touch gestures based biometric authentication scheme for touchscreen mobile phones. In *the Information Security and Cryptology*. Springer, 331–350.

[18] George D Montañez, Ryen W White, and Xiao Huang. 2014. Cross-device search. In *the Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 1669–1678.

[19] YC Hacker News. 2017. (2017). https://news.ycombinator.com/.

[20] OFCOM. 2012. (2012). http://media.ofcom.org.uk/news/2012/uk-is-now-texting-more -than-talking/.

[21] Alexander P Pons and Peter Polak. 2008. Understanding user perspectives on biometric technology. *the Communications of the ACM* 51, 9 (2008), 115–118.

[22] HackerNews Reader. 2017. (2017). https://play.google.com/store/apps/details?id=com.xw.hackernews&hl=en.

[23] Napa Sae-Bae, Nasir Memon, Katherine Isbister, and Khandakar Ahmed. 2014. Multitouch gesture-based authentication. *the IEEE Transactions on Information Forensics and Security* 9, 4 (2014), 568–582.

[24] Chao Shen, Zhongmin Cai, and Xiaohong Guan. 2012. Continuous authentication for mouse dynamics: A pattern-growth approach. In *the 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 1–12.

[25] Emmanuel Munguia Tapia, Stephen S Intille, and Kent Larson. 2004. Activity recognition in the home using simple and ubiquitous sensors. In *the International Conference on Pervasive Computing*. Springer, 158–175.

[26] Dirk Van Bruggen, Shu Liu, Mitch Kajzer, Aaron Striegel, Charles R Crowell, and John D'Arcy. 2013. Modifying smartphone user locking behavior. In *the Proceedings of the Ninth Symposium on Usable Privacy and Security*. ACM, 10.

[27] Tam Vu, Ashwin Ashok, Akash Baid, Marco Gruteser, Richard Howard, Janne Lindqvist, Predrag Spasojevic, and Jeffrey Walling. 2012. Demo: user identification and authentication with capacitive touch communication. In *the Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*. ACM, 483–484.

[28] Yu Wang, Xiao Huang, and Ryen W White. 2013. Characterizing and supporting cross-device search tasks. In *the Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*. ACM, 707–716.

[29] Hui Xu, Yangfan Zhou, and Michael R Lyu. 2014. Towards continuous and passive authentication via touch biometrics: An experimental study on smartphones. In *the Symposium On Usable Privacy and Security*, Vol. 14. 187–198.

[30] Tong Yu, Yong Zhuang, Ole. Mengshoel, and Osman. Yagan. 2016. Hybridizing Personal and Impersonal Machine Learning Models for Activity Recognition on Mobile Devices. In *the Proceedings of the 8th International Conference on Mobile Computing, Applications and Services (MobiCASE-16)*. Cambridge, Great Britain.

[31] Ming Zeng, Xiao Wang, Le Nguyen, Pang Wu, Ole. Mengshoel, and Joy. Zhang. 2014. Adaptive activity recognition with dynamic heterogeneous sensor fusion. In *the Proceedings of the 6th International Conference on Mobile Computing, Applications and Services (MobiCASE-14)*. Austin, TX, 189–196.

[32] Jiang Zhu, Pang Wu, Xiao Wang, and Juyong Zhang. 2013. Sensec: Mobile security through passive sensing. In *the International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 1128–1133.