Fingerprint Attack against Touch-enabled Devices

Yang Zhang
Southeast University
Nanjing 211189, P.R.China
yangzhang@seu.edu.cn

Peng Xia
UMass Lowell
Lowell, MA 01854, USA
pxia@cs.uml.edu

Junzhou Luo
Southeast University
Nanjing 211189, P.R.China
jluo@seu.edu.cn

Zhen Ling
Southeast University
Nanjing 211189, P.R.China
zhenling@seu.edu.cn

Benyuan Liu
UMass Lowell
Lowell, MA 01854, USA
bliu@cs.uml.edu

Xinwen Fu
UMass Lowell
Lowell, MA 01854, USA
xinwenfu@cs.uml.edu

ABSTRACT

Oily residues left by tapping fingers on a touch screen may breach user privacy. In this paper, we introduce the fingerprint attack against touch-enabled devices. We *dust* the touch screen surface to reveal fingerprints, and use an iPhone camera to carefully photograph fingerprints while striving to remove the virtual image of the phone from the fingerprint image. We then sharpen the fingerprints in an image via a various image processing techniques and design effective algorithms to *automatically* map fingerprints to a keypad in order to infer tapped passwords. Extensive experiments were conducted on iPad, iPhone and Android phone and the results show that the fingerprint attack is effective and efficient in inferring passwords from fingerprint images. To the best of our knowledge, we are the first using fingerprint powder on touch screen and inferring passwords from fingerprints. Video at `http://www.youtube.com/watch?v=vRUbJIcV9vg` shows the dusting process on iPhone and video at `http://www.youtube.com/watch?v=6jS6KroER3Y` shows the dusting process on iPad. After dusting, password characters for login are clearly disclosed.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection; K.4.1 [**Public Policy Issues**]: Abuse and Crime Involving Computers

General Terms

Security

Keywords

Fingerprint, Touch Screen, iPhone, iPad, Android, Attack

1. INTRODUCTION

Touch-screen devices enjoy increasingly popular usage. Displaybank forecasts that the total touch-screen panel market size will grow to $9.65 billion and 1.35 billion units, and 800 million smartphones are expected to be touch-enabled in 2014 [14]. The market of smartbooks including tablets with touch screen will catch up with the market of mobile phones. Touch screens are also widely used in consumer-electronic products such as personal digital assistants (PDAs), personal navigation devices, netbooks and laptops.

However, oily residues (i.e. smudges) left by tapping fingers on a touch screen may disclose a plethora of information about its owner/users. Aviv *et al.* [2] introduced the smudge attack, which explores graphical passwords used by an Android smartphone. A graphical password corresponds to a graphical pattern when a finger presses on a touch screen and drags around. They studied how to *directly* take pictures of the smudges from a smartphone touch screen surface at various camera angles with respect to the orientation of the phone and showed that smudge attack can reveal the graphical pattern-based passwords in Android.

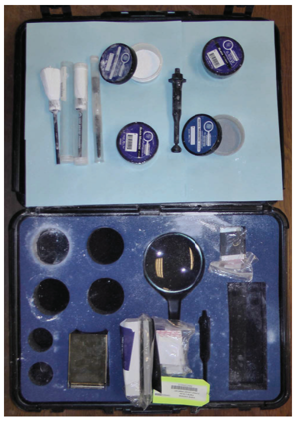


Figure 1: Standard Latent Print Field Kit

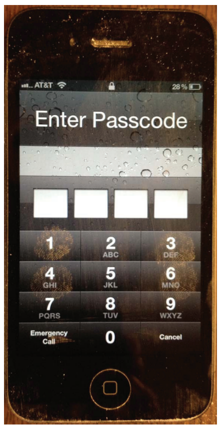


Figure 2: Fingerprint on iPhone

In this paper, we introduce a fingerprint attack against tapped passwords via a keypad instead of graphical passwords. In this attack, an attacker first dusts the touch screen with fingerprint powder to reveal fingerprints left from tapping fingers. She then photographs the fingerprints, maps the fingerprints to the on-screen keyboard and recovers the password characters. Brute force methods can then be applied to derive the actual password sequence. For a 4-characters PIN like those used by iPhone and iPad, an attacker just needs to try 12 times on average to break into the phone and read the open Gmail and collect sensitive information from other applications such as phone book. Fingerprints are often not visible and smudge attack in [2] cannot work in these scenarios. Fingerprinting kits are also widely available and a standard latent print field kit costs only $30 [9] while we used a professional latent fingerprint kit in Figure 1 [8], which costs $200. The professional latent fingerprint kit contains a set of black and white fingerprint powder and dusting brushes as well as a set of magnetic powder and magnetic applicator, which were not used in our experiments.

Figures 2 and 3 show the fingerprints on iPhone and iPad after dusting and the photos are taken with iPhone 4s. We trimmed the background and kept only iPhone and iPad images. Our video at `http://www.youtube.com/watch?v=vRUbJIcV9vg` shows the dusting process on iPhone and video at `http://www.youtube.com/watch?v=6jS6KroER3Y` shows the dusting process on iPad. We can see that the password characters for login are clearly disclosed. A fiberglass brush and white powder are used for dusting and revealing the fingerprints on touch screen.

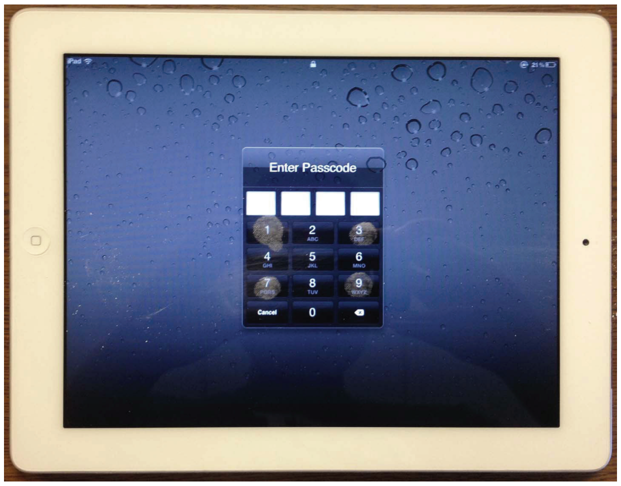


Figure 3: Fingerprint on iPad

Our **threat model** is as follows [10]: the attacker has physical access to a touch-enabled device. This is a reasonable assumption in many scenarios. An "attacker" such as a spouse who has physical access to a smartphone can apply fingerprint powder to a smartphone and deploy the attack. Corrupt staff at a working place deployed with touch-enabled devices may also wield a fingerprinting kit and collect "passwords on screens".

The major contributions of this paper are summarized as follows:

- We conducted a systematic study of inferring a password from photographed fingerprint images. We have investigated various practical issues such as selecting approbate fingerprint powder for dusting, removing virtual images of the phone camera during photographing, sharpening fingerprints via various image processing techniques, designing algorithms to automatically infer passwords from fingerprint images, and *differentiating fingerprints from multiple persons sharing a touch-enabled device*.
- Extensive experiments on iPad, iPhone and Android phone were performed to verify the feasibility and effectiveness of the fingerprint attack against touch-enabled devices. In most scenarios, the attack can reveal more than 50% of the passwords. We were also able to differentiate fingerprints from people sharing a device.

The rest of the paper is organized as follows: Section 2 introduces most related work. In Section 3, we introduce the fingerprint attack to infer the password from fingerprint images. We evaluate the attack in Section 4 and briefly discuss countermeasures in Section 5. Section 6 concludes this paper.

2. RELATED WORK

Felt *et al.* [10] classify threats from third-party smartphone applications into *malware*, *grayware*, and *personal spyware*. Malware intends to damage finance or property of the smartphone owner. An "attacker" such as a spouse who has physical access to a smartphone can install personal spyware on the victim smartphone and gather information about the smartphone owner, for example, tracking the victim. Grayware is often commercial applications with real functionality while stealing user information. The distributor may have a privacy policy with varying degree of clarity. The authors conduct a survey of 46 pieces of smartphone malware and their incentives and conclude that Apple's mechanisms of application permission and review process can avoid approving malware. Becher *et al.* [5] examine mechanisms securing sophisticated mobile devices. Although no major incidents of attacking smartphones have happened, small-scale attacks have been emerging. Threats are classified into four classes: *hardware centric*, *device independent*, *software centric*, and *user layer attacks* for the purpose of eavesdropping, availability attacks, privacy attacks and impersonation attacks. Existing security mechanisms are enumerated for various attacks.

TouchLogger [6] is an Android malware, which is installed by a victim and utilizes device orientation data to infer keystrokes. When a user types on a virtual keyboard on a touch screen, the orientation event reports intrinsic Tait-Bryan angles and timing and reflects device orientation, which is user independent in terms of typed keys. TouchLogger infers the typing locations from Tait-Bryan angles and timing information and derives the corresponding keys. Owusu *et al.* show [22] that a malware can use only accelerometer data to infer the entered keys on a virtual keyboard. The inference accuracy is constrained by the sampling frequency of the accelerometer, the key location, and its size. Pattern recognition is used and 46 features are generated from each preprocessed acceleration stream. TapLogger [30] also uses motion sensors to infer a user's tap inputs to a smartphone.

In [23], reflections of a device's screen on a victim's glasses or other objects are exploited to automatically infer text typed on a virtual keyboard. The authors use inexpensive cameras (such as those in smartphones), utilize the fact of keys popping out when pressed and adopt computer vision techniques processing the recorded video in order to infer the corresponding key although the text in the video is illegible. Balzarotti *et al.* [4] proposed an automatic approach to reconstruct the text typed on a keyboard from a video of a person typing on a physical keyboard. They assume that the attacker can deploy a camera to record the victim's hand on the keyboard, and the camera has a static and clear view of the typing hand on the keyboard. Computer vision analysis is applied to analyze each frame of the video and reconstruct the keys pressed by the victim. Maggi *et al.* [15] implemented an automatic shoulder-surfing attack against touch-enabled mobile devices. In their work, the attacker can employ a video camera to the target screen when the victim inputs the text on a touch screen. Then the system processes the stream of images frame by frame in order to detect the touch screen, rectify and magnify the screen images, and ultimately identify the popping up keys typed by the user.

In [17], an iPhone is used to sense vibrations via accelerometers from a nearby keyboard and infer the entered text. The challenge is that the sampling rates of the accelerometer running in a modern mobile phone is often very low, merely 100 Hz. The authors use profiles of pairs of keypress events and neural networks to recover the text from barren accelerometer data. Keystrokes can also be derived from acoustic frequency signatures [1], timings between two keystrokes [13], and statistical constraints of English language from sound recordings [32]. Electromaganetic emanation of keyboards is also studied for keylogging [29].

Michal Zalewski [31] takes advantage of the thermal residue of finger left on the pressed keys on a keypad in order to infer typed keys. Since the thermal residue on the keypad will persist within up to approximately five to ten minutes, the attacker can approach the keypad after the password was entered and use a thermal imaging camera to detect the individual keys tapped by the user. Mowery *et al.* [19] analyzed the effectiveness of this attack from three separate aspects, including keypad surface materials, the diversity of body heat between people using the keypads, and the scalability of the attack. The order of the password characters can be deduced from which key is "hotter".

This paper addresses a problem different from the smudge attack [2]. Our fingerprint attack targets tapped passwords via a keypad instead of graphical passwords in [2].

3. FINGERPRINT ATTACK AGAINST TOUCH SCREEN

In this section, we present the basic idea of the fingerprint attack and elaborate the detailed workflow of the attack, including preprocessing, preserving fingerprints and mapping fingerprints to passwords.

3.1 Basic idea

The password tapped by a user on a touch-enabled device can be inferred through the fingerprints left on the surface of the touch screen, and this causes the user's privacy leakage. Due to residues of oils left in the shape of the friction ridges, the impressions of a user's fingerprints can be left behind on the surface of the touch screen after the user tapped the touch screen. We can identify the relevant positions of fingerprints on the screen and infer the specific tapped keys in order to deduce the user's password.

Figure 4 illustrates the basic workflow of the fingerprint attack to reveal the user's password from fingerprints left on a touch screen. We take iPad as an example to explain the basic idea.

- *Preprocessing*: an attacker dusts the surface of iPad with appropriate fingerprint powder to reveal the detail of the fingerprints on the touch screen.
- *Preserving fingerprints*: a camera such as an iPhone camera is used to take two photographs, the fingerprint image and the keypad image by turning off/on the backlight. The reason for using the two-photographs strategy is that it caters to various scenes. For example, a touch screen can be full of fingerprints. By taking photographs, the attacker can analyze them carefully at home.
- *Mapping fingerprints to keypad*: the attacker maps the fingerprints in one image to the keypad in the other image and recover the tapped keys.

The upper half of Figure 4 shows a pragmatic example of the workflow. A user taps numeric keys {2,4,6,8} on iPad to unlock the screen. An attacker physically approaches the iPad, preprocesses and preserves the fingerprints. Finally, the attacker compares the two images and maps the positions of fingerprints to positions of keys on the keypad in order to recover the tapped keys for unlocking the screen. Note: the gray areas in the far right image at the upper half of Figure 4 are excluded keys.

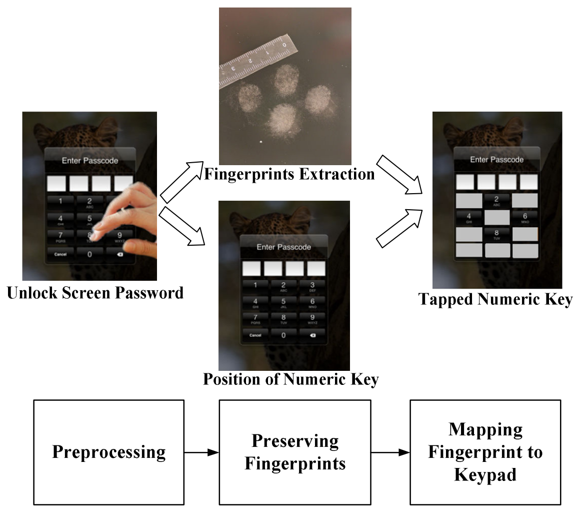


Figure 4: Workflow of the Fingerprint Attack

In the rest of this section, we will discuss these three steps in detail and discuss the problem of differentiating fingerprints from multiple persons sharing a device at the end of this section.

3.2 Preprocessing

Ridges in human skin are responsible for the existence of fingerprints and oily residues of tapping fingers on a touch screen may not always produce visible fingerprints [18]. In our scenario, we want to retrieve fingerprints with enough ridge details so that we may differentiate fingerprints from multiple persons sharing a device and ease the difficulty of inferring passwords. Therefore, we adopt the dusting strategy used in a crime scene investigation and apply fingerprint powder to a touch screen surface in order to reveal the fingerprints.

We need to carefully choose fingerprint powder for quality fingerprints and ease of photographing. Two principles are developed through our experiments for dusting these mirror-like touch screens.

- First, different powders may only be good for specific environments. In our case, a variety of fingerprint powders can be used on the smooth and dry touch screen surfaces, including aluminum powder, bronze powder, cupric oxide powder, iron powder, titanium dioxide powder, graphite powder, magnetic powder, and fluorescent powder.
- Second, the powder should be in best contrast with the background. Accordingly, we choose bronze or white powder, the most common fingerprint powder used on a dark background for the contrast effect.

3.3 Preserving Fingerprints

After applying the fingerprint power, we should be able to photograph visible fingerprints on the touch screen. Since we will design algorithms to automatically map fingerprints in a fingerprint image to a keypad, reflections such as the camera itself is not desirable in fingerprint images. We found that with appropriate lighting, digital single-lens reflex (DSLR) cameras can almost completely remove those undesirable reflections. Figure 5 shows the operation platform with a DSLR camera mounted on the top of the frame. Figure 6 shows a set of high-quality fingerprints on iPad, taken by a DSLR camera.

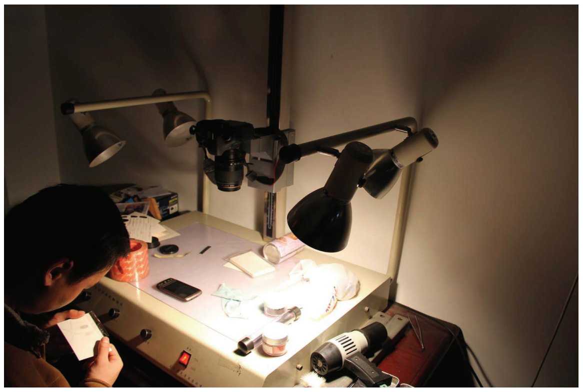


Figure 5: Operation Table with a DSLR Camera

Since a camera phone is more portable and convenient than DSLR, we select the camera phone such as the iPhone 4 camera to take photographs. With appropriate photographing and imaging processing techniques, most reflections can also be removed and clear fingerprints can be photographed with such a common-place camera, as shown by our experiments in Section 4.

3.3.1 *Taking Photographs*

Once the visible fingerprints can be observed after dusting, we first turn off the backlight of iPad and choose the camera phone (or digital single-lens reflex (DSLR) camera)

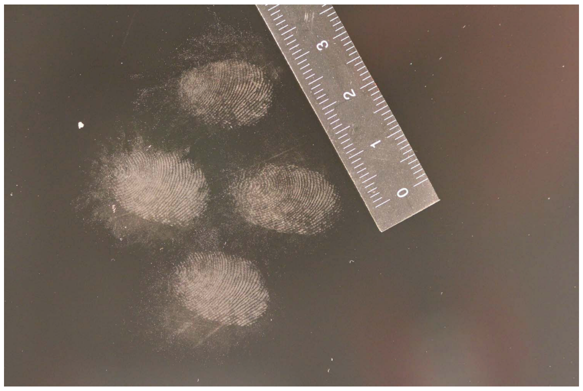


Figure 6: Fingerprint on iPad Taken by a DSLR Camera

to take photographs of the fingerprints. We now present the principle of taking fingerprint photographs, and then discuss the practical issues and our solutions.

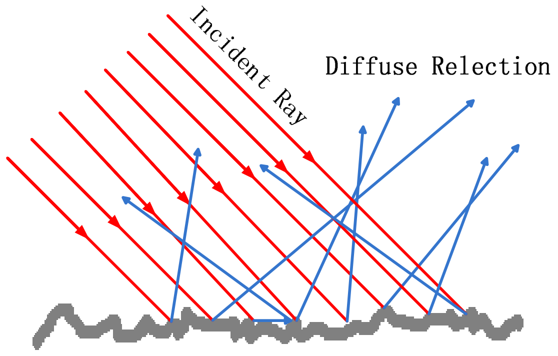


Figure 7: Diffuse Reflection

To better understand the image formation of the fingerprints on the camera, we investigate the diffuse reflection of the fingerprint as shown in Figure 7. The visibility of objects is primarily caused by diffuse reflection of light and the diffusely-scattered light forms the image of the object in the observer's eye [16]. In diffuse reflection, when *incident rays* hit a surface, they are reflected at many angles. In *specular reflection*, incident rays are reflected at one angle. The surface of many common materials, such as the touch screen, exhibits a mixture of diffuse and specular reflection. Since the diffuse reflection of light from the object forms the image of the object in the observer's eye, we can observe the image of the object. By contrast, if the light of an object is specularly reflected on the surface and diverged, the virtual image of the object in the observer's eyes appears to converge in or behind the surface.

To derive clear fingerprint photographs, the optical component of a camera should receive more diffuse reflection of light from the fingerprint. The powder brushed on the surface of iPad can enhance the diffuse reflection of the fingerprints. We can also use strong light to better illuminate the fingerprints for better diffuse reflection and clearer fingerprint photographs.

After preserving the fingerprint, we should keep the posi-

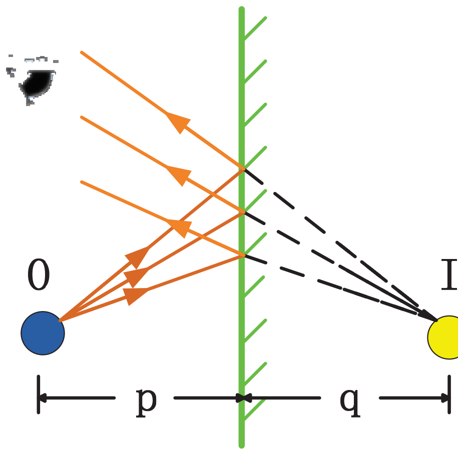


Figure 8: Specular Reflection

tion of our camera and turn on the iPad to take a picture of the keypad in order to accurately map the fingerprints to the tapped keys.

3.3.2 *Practical Issues and Solutions*

In practice, as a result of the specular reflection on the surface, the reflection of the camera on the screen can also be photographed. In this case, the virtual image of the camera and image of fingerprints will mix together and it is difficult to identify the clear fingerprint from the photograph. To deduce correct password characters, we need to eliminate the virtual image of the camera. Figure 8 shows how the a virtual image forms. Assume a point source of light is placed at O with a distance p in front of the surface. Light rays leave the source O and are reflected from the surface and continue to diverge, but they appear to us to come from a point I behind the mirror. We always locate images by extending diverging rays back to a point from which they appear to diverge. Because the rays in Figure 8 appear to originate at I, which is a distance q behind the mirror, then we conceive point I as the location of the image. This type of object image is called *virtual image*.

According to the principle of forming a virtual image on the touch screen, a camera will photograph the virtual image of itself if the specular reflection rays from the camera are captured by the optical component of the camera. Intuitively, we can keep the camera from being illuminated in order to avoid forming a virtual image on the screen. However, in practice, it is difficult to evade illuminating the camera and the virtual image of the camera on the photograph.

We can adopt photographing techniques to eliminate the virtual image of the camera as below:

- Autofocus by luminance contrast. Contrast detection is a common technique used in photographing. With autofocus, the camera adjusts its contrast method in focusing an object, senses luminance from all directions and autofocuses on the brightest point. In the worst case that inadequate light is provided, we can use a LED flash built into the camera to produce a flash of artificial light, and the luminance of reflected flash light rays on the screen is stronger than that of the reflected light rays from the camera. By using the contrast method, the camera can autofocus on the brighter part, i.e., fingerprints, illuminated by the LED flash rather than the virtual image of camera. Thus, we can eliminate the camera virtual image. [16]
- Depth of Field (DOF). When a lens focuses on a distance, objects within a range will appear sharp to our eyes. Depth of field is the distance between the nearest and farthest objects that appear sharp in a scene [7]. Beyond DOF, objects will appear blurry. In order to make the fingerprints sharp and the camera virtual image blurry, we can select an appropriate small DOF in order to de-emphasize the virtual image. If the distance between the virtual image and the camera exceeds the camera's focus distance, the virtual image can be de-emphasized so that we can avoid photographing a clear image of the camera, but derive a clear fingerprint photograph.

As mentioned before, we can choose either a camera phone or DSLR to photograph fingerprints. As a matter of fact, the camera phone can only use the first technique to de-emphasize the virtual image, while DSLR can use both techniques. However, according to our experiment in Section 4, we can obtain clearly visible fingerprints using the camera phone. Since the camera phone is more portable and convenient than DSLR, we select the camera phone to take photographs. We also use various image processing techniques to remove the background including the camera virtual image from the fingerprint photograph.

3.4 Mapping Fingerprints to Keypad

We first introduce the basic idea of mapping fingerprints to a keypad and then discuss the selection of thresholds to correctly recognize all the password characters.

Algorithm 1 Fingerprints Image Processing Algorithm

```
Require: An image with fingerprints
1: Resize original image into $800 \times 600$ pixels;
2: Convert resized image into grayscale image;
3: Apply Laplace edge detection algorithm [27] to grayscale
   image to sharpen fingerprints and eliminate the background;
4: Apply maximum between-class variance method [21] to
   derive binary image: Fingerprint is in white and
   background is in black;
5: Divide binary image into $20 \times 20$ pixel grids;
6: Construct fingerprint matrix $A = \{a_{ij} | 0 \le i < 40, 0 \le j < 30\}$ where $a_{ij}$ is the number of white points in the
   grid at column $i$ and row $j$.
```

3.4.1 *Basic Idea of Mapping*

To automatically retrieve password characters, we first process the fingerprint and keypad images (taken at the same position) and derive what areas (represented by a matrix) fingerprints and keys (of a keypad) occupy in the corresponding images. In our algorithms in this paper, we often take iPad as an example and algorithms for other devices are similar. Algorithm 1 shows how we extract fingerprints, and construct fingerprint matrix A after converting the fingerprint photo into a binary image and dividing it into grids. Each number in matrix A stands for the number of white points in each grid. The larger area a fingerprints occupies,

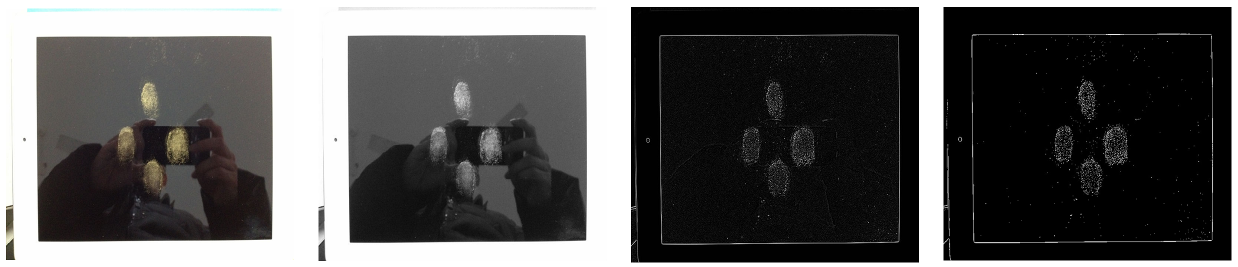
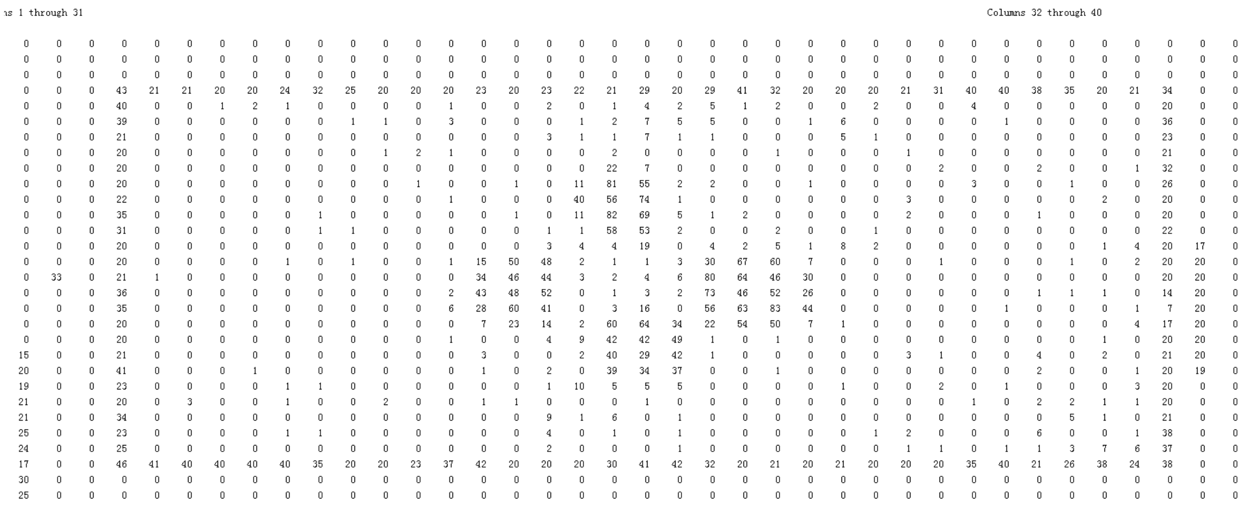

(a) Step one (b) Step two (c) Step three (d) Step four

Figure 9: An Example of Fingerprints Image Processing



```
ns 1 through 31                                                                                                   Columns 32 through 40

   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
   0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
   0    0    0   43   21   21   20   20   24   32   25   20   20   20   23   20   23   22   21   29   20   29   41   32   20   20   20   21   31   40   40   38   35   20   21   34    0    0
   0    0    0   40    0    0    1    2    1    0    0    0    0    1    0    0    2    0    1    4    2    5    1    2    0    0    2    0    0    4    0    0    0    0    0   20    0    0
   0    0    0   39    0    0    0    0    0    0    1    1    0    3    0    0    0    1    2    7    5    5    0    0    1    6    0    0    0    0    1    0    0    0    0   36    0    0
   0    0    0   21    0    0    0    0    0    0    0    0    0    0    0    0    3    1    1    7    1    1    0    0    0    5    1    0    0    0    0    0    0    0    0   23    0    0
   0    0    0   20    0    0    0    0    0    0    0    1    2    1    0    0    0    0    2    0    0    0    0    1    0    0    0    1    0    0    0    0    0    0    0   21    0    0
   0    0    0   20    0    0    0    0    0    0    0    0    0    0    0    0    0    0   22    7    0    0    0    0    0    0    0    0    2    0    0    2    0    0    1   32    0    0
   0    0    0   20    0    0    0    0    0    0    0    0    1    0    0    1    0   11   81   55    2    2    0    0    1    0    0    0    0    3    0    0    1    0    0   26    0    0
   0    0    0   22    0    0    0    0    0    0    0    0    0    1    0    0    0   40   56   74    1    0    0    0    0    0    0    3    0    0    0    0    0    2    0   20    0    0
   0    0    0   35    0    0    0    0    0    1    0    0    0    0    0    1    0   11   82   69    5    1    2    0    0    0    0    2    0    0    0    1    0    0    0   20    0    0
   0    0    0   31    0    0    0    0    0    1    1    0    0    0    0    0    1    1   58   53    2    0    0    2    0    0    1    0    0    0    0    0    0    0    0   22    0    0
   0    0    0   20    0    0    0    0    0    0    0    0    0    0    0    0    3    4    4   19    0    4    2    5    1    8    2    0    0    0    0    0    0    1    4   20   17    0
   0    0    0   20    0    0    0    0    1    0    1    0    0    1   15   50   48    2    1    1    3   30   67   60    7    0    0    0    1    0    0    0    1    0    2   20   20    0
   0   33    0   21    1    0    0    0    0    0    0    0    0    0   34   46   44    3    2    4    6   80   64   46   30    0    0    0    0    0    0    0    0    0    0   20   20    0
   0    0    0   36    0    0    0    0    0    0    0    0    0    2   43   48   52    0    1    3    2   73   46   52   26    0    0    0    0    0    0    1    1    1    0   14   20    0
   0    0    0   35    0    0    0    0    0    0    0    0    0    6   28   60   41    0    3   16    0   56   63   83   44    0    0    0    0    0    1    0    0    0    1    7   20    0
   0    0    0   20    0    0    0    0    0    0    0    0    0    0    7   23   14    2   60   64   34   22   54   50    7    1    0    0    0    0    0    0    0    0    4   17   20    0
   0    0    0   20    0    0    0    0    0    0    0    0    0    1    0    0    4    9   42   42   49    1    0    1    0    0    0    0    0    0    0    0    0    1    0   20   20    0
  15    0    0   21    0    0    0    0    0    0    0    0    0    0    3    0    0    2   40   29   42    1    0    0    0    0    0    3    1    0    0    4    0    2    0   21   20    0
  20    0    0   41    0    0    0    1    0    0    0    0    0    0    1    0    2    0   39   34   37    0    0    1    0    0    0    0    0    0    0    2    0    0    1   20   19    0
  19    0    0   23    0    0    0    0    1    1    0    0    0    0    0    0    1   10    5    5    5    0    0    0    0    1    0    0    2    0    1    0    0    0    3   20    0    0
  21    0    0   20    0    3    0    0    1    0    0    2    0    0    1    1    0    0    0    1    0    0    0    0    0    0    0    0    0    1    0    2    2    1    1   20    0    0
  21    0    0   34    0    0    0    0    0    0    0    0    0    0    0    0    9    1    6    0    1    0    0    0    0    0    0    0    0    0    0    0    5    1    0   21    0    0
  25    0    0   23    0    0    0    0    1    1    0    0    0    0    0    0    4    0    1    0    1    0    0    0    0    0    1    2    0    0    0    6    0    0    1   38    0    0
  24    0    0   25    0    0    0    0    0    0    0    0    0    0    0    0    2    0    0    0    1    0    0    0    0    0    0    1    1    0    1    1    3    7    6   37    0    0
  17    0    0   46   41   40   40   40   40   35   20   20   23   37   42   20   20   20   30   41   42   32   20   21   20   21   20   20   20   35   40   21   26   38   24   38    0    0
  30    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
  25    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
```

Figure 10: Result of Fingerprints Image Processing

the larger the number is in the matrix. Algorithm 2 uses similar procedures to derive the mapping between keys (k_m) and grids. A grid set K_m records the set of grids that k_m occupies and is the area where k_m is located.

Figure 9 shows an example of practically using Algorithm 1 to process a fingerprint image. Four subpictures represent the results of Step 1 to Step 4 in the algorithm respectively. Figure 10 represents the number of white dots in each grid according to Step 5 and Step 6. It is obvious that the number of white dots in certain regions where a user has tapped is much larger than that of other regions. Thus, we have extracted the position of fingerprints on the touch screen.

Figure 11 shows an example of pragmatically using Algorithm 2. Three subpictures represent the results of Step 1 to Step 3 in the algorithm respectively. We found that the unlocking screen password dialog box of Mac IOS are of the same form and size, and hence all areas are of fixed length-width ratio, as shown in Figure 12. The numeric digits in Figure 12 show the size of length, width, left margin and right margin of the input boxes and keys. The unit of length in Figure 12 is centimeter. Based on this ratio, we used relative position to locate each key of the soft keyboard. Since white input boxes have obvious characteristics as shown in Figure 11 (c), we first figure out location, length, width, left margin and right margin of four input boxes by scanning the picture to determine the grids of the four white input boxes.

Algorithm 2 Keypad Image Processing Algorithm

```
Require: An image with keypad
1: Resize original image into $800 \times 600$ pixels;
2: Convert resized image into grayscale image;
3: Apply minimum error method [3] to grayscale image and derive binary image;
4: Derive the area $Z(k_m)$ corresponding to key $m$ in the binary image, where $0 \leq m \leq 9$;
5: Divide binary image into $20 \times 20$ pixel grids and derive grid matrix $B = \{ b_{ij} \mid 0 \leq i \leq 40,\ 0 \leq j \leq 30 \}$ where $b_{ij}$ is the grid at column $i$ and row $j$ ;
6: Derive grid set $K_m$ for key $m$, $K_m = \{b_{pq} \mid 0 \leq p < 40, 0 \leq q < 30, b_{pq} \in Z(k_m)\}$
```

Second, according to the grids of the four white input boxes and the ratio of unlocking screen password dialog box, we are capable of figuring out the grids of each key.

With fingerprint matrix A and a *key*'s grid set K_m, we can map a fingerprint to a key k_m. We can derive the total number of white points (corresponding to a fingerprint) N_m in a *key* area K_m. We sort the sequence $N = \{N_0, N_1, \ldots, N_9\}$ in the decreasing order. Recall that we may not know the password length and the number of repeating characters in a password. We propose to select candidate keys from the sequence N based on a threshold and will discuss the thresh-

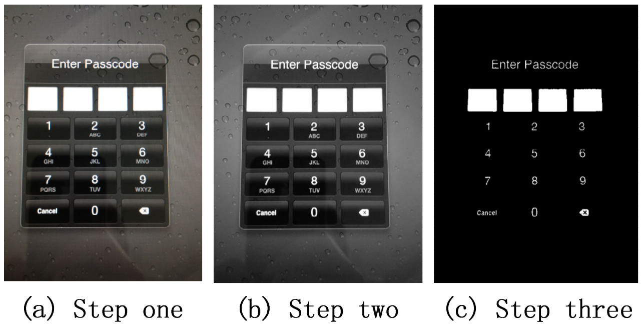


Figure 11: An Example of Keypad Image Processing

old selection below. Astute readers may come up with this question: the border of a numeric key may not match well with that of girds, and that is to say a numeric key may not occupy natural numbers of girds. The problem can be solved by dividing the original picture into smaller grids. For example, we divide a 800×600 pixels picture into 4×4 pixels grids which are of 200 columns and 150 rows. The extreme case is that each pixel in the picture represents a grid.

3.4.2 Detection Rate and Threshold Selection

Our evaluation metric for inferring a password from fingerprints is *detection rate*, the probability that only and all characters of the password are recovered from fingerprints. Our problem is: given $N = \{N_0, N_1, \ldots, N_9\}$, how can we select an appropriate approach to correctly recognize password characters? Recall that N_i ($0 \leq i \leq 9$ in the case of iPad and iPhone's unlocking password) is the total number of white points in a *key* area K_m, and fingerprint powder may be sprayed onto the whole keypad and white points (supposedly corresponding to pressed keys) may show up in un-pressed key areas.

Clustering Based Approach. Intuitively, we can use a clustering algorithm to separate tapped keys from untapped keys. We select the well-known clustering algorithm, K-means cluster algorithm. The K-means clustering algorithm can classify the keys, i.e., $N = \{N_0, N_1, \ldots, N_9\}$, into two categories: the tapped keys and untapped keys, that is, $K = 2$. In our experiments, we observed that the more keys pressed, the smaller detection rate of using the K-means algorithm to cluster correctly. Therefore, the k-means clustering algorithm is not a good choice in our case.

Threshold Based Approach. Given a low detection rate by the K-means algorithm, we propose a threshold based approach to automatically identify the tapped keys on the screen. We first need to determine a threshold, *press-threshold*, to recognize pressed keys. Press-threshold λ_m for a key k_m is defined as follows,

$$\lambda_m = \frac{N_m - \min_{i=0}^{9} N_i}{\max_{i=0}^{9} N_i - \min_{i=0}^{9} N_i}. \qquad (1)$$

Equation (1) derives a normalized threshold, considering that different people have different habits tapping on a keypad. Again, here we use a 10-digits keypad as the example. However, our approach can be extended to other cases straightforward. Our experiments actually examined general software keyboards.

Figure 13 shows how we derive the press-threshold. We set press-threshold from 0 to 1 with a fixed step length of 0.01.

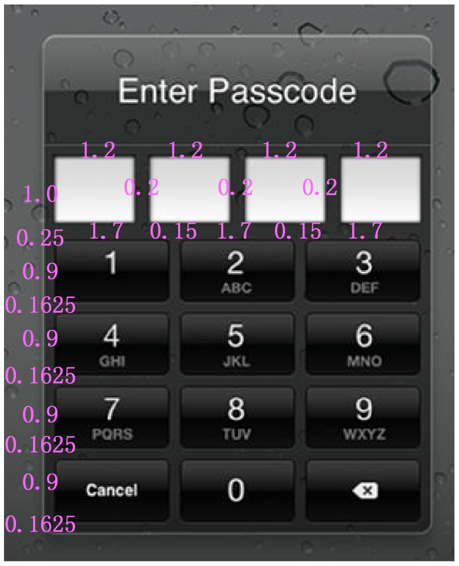


Figure 12: Length-width Ratio of a standard unlocking Screen Password Dialog Box of Mac IOS

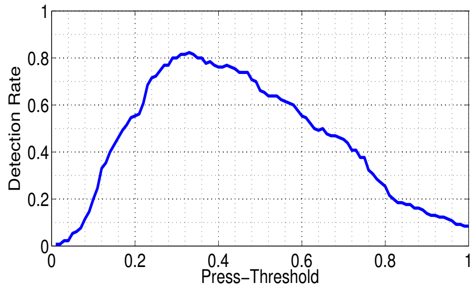


Figure 13: Deriving Press-threshold

The detection rate of each press-threshold is shown in Figure 13 where the press-threshold corresponding to the highest detection rate is the optimized one. In our experiments, we found that a press-threshold of 0.33 produces an optimized detection rate for pressed keys.

A key may be pressed multiple times and we also need to select the *overlap-threshold* and determine what keys are pressed multiple times, after pressed keys are already recognized by the press-threshold. For example, if a password length is 4, after determining that 3 keys have been pressed by using press-threshold, we can figure out that the key with maximum N_i is the repeating key. If we find that 2 keys have been pressed, we need to determine the password has a 2-2 pattern (i.e. two keys are pressed and each is pressed two times) or 3-1 pattern. The *overlap-threshold* is defined as follows,

$$\Phi = \frac{max_{i=0}^{9} N_i - min_{i=0}^{9} N_i}{second\_max_{i=0}^{9} N_i - min_{i=0}^{9} N_i} \qquad (2)$$

Figure 14 shows how we derive the overlap-thresholds. We set the overlap-threshold from 0 to 1 with fixed step length of 0.01. Figure 13 shows the detection rate in terms of overlap-threshold for determining whether the key has

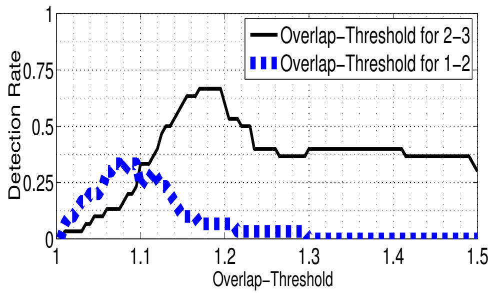


Figure 14: Deriving Overlap-threshold

been pressed once or twice and detection rate in terms of overlap-threshold for determining whether the key has been pressed twice or three times. The overlap-thresholds corresponding to the highest detection rate are the optimized overlap-thresholds we want. In our experiments, we find that overlap-threshold for determining whether the key has been pressed once or twice is 1.19. Overlap-threshold for determining whether the key has been pressed twice or three times is 1.39.

3.5 Differentiating Fingerprints from Multiple Persons Sharing a Device

In case that a touch-enabled device is shared with multiple people, it is necessary to confirm that fingerprints in the sensitive region such as the area of keypad belong to the same person. We may also classify those fingerprints into groups, each of which belongs to one person. This can ease the effort of inferring the password.

Depending on the size of the on-screen keyboard and individual habit, people may use either **finger tip** or a **whole finger** (whose core is kept in the fingerprint). We have tested both types of fingerprints in our experiments. Fingerprints consist of pattern of ridges and valleys on the surface of a finger [18]. Figure 15 illustrates the ridges and valleys of a fingerprint. The white lines (formed by fingerprint powder) are ridges, between which black space is valleys. Figure 15 also shows three basic patterns of fingerprint ridges, i.e., arch, loop and whorl. Each fingerprint can be determined by the overall structure (arch, loop and whirl) and local ridge structure called *minutiae points* such as a ridge bifurcation or a ridge ending. Algorithms for fingerprint matching are based on either minutiae or global structures and local landmarks. Minutiae are ridge ending and ridge bifurcation and algorithms based on minutiae utilize relative positions of minutiae pairs.

We have tested three algorithms to match and classify fingerprints: filterbank [12, 26], adjacent orientation vector (AOV) [11, 24] and correlation-filter [28, 25]. The filterbank approach requires the local core of each fingerprint and is not good for matching fingerprints without cores such as finger tips. The AOV approach uses the number of possible local minutiae pairs to match fingerprints. Since AOV does not capture the global structure, its performance depends on the detailed information of the fingerprints. The correlation-filter approach uses fast Fourier transform to catch both global and local structures. A more detailed discussion of these three techniques is given below.

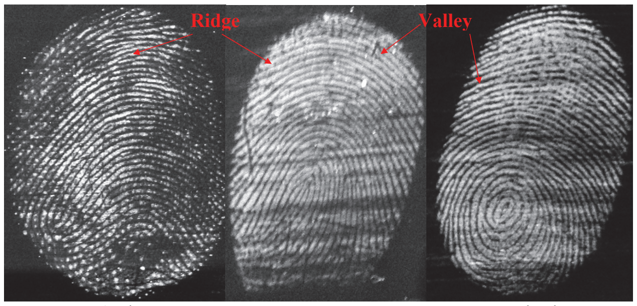


Figure 15: Arch, Loop and Whorl

- Filterbank-based approach utilizes both local ridge and valley anomalies and global patterns. It first selects a reference point and the region of interest around the reference point in the fingerprint image, divides the region of interest into sectors, and normalizes each sector to remove noise caused by optical sensors and gray-level deformation. A bank of Gabor filters are then used to filter each sector in eight different directions and transform local discriminatory information in each sector into biorthogonal components in terms of spatial frequencies. In this way, a feature vector is formed by collecting all the local features and is called *FingerCode*. The simple Euclidean distance between FingerCodes can be used to match two images. Refer to [26] for an example implementation of the filterbank-based approach.

- Adjacent orientation vector system uses only minutiae pairs in the fingerprint matching process. In this approach, all ridges are systematically numbered. AOV (adjacent orientation vector) is a vector consisting of the difference between adjacent minutiae orientations, and ridge number of adjacent minutiae pairs. To match two fingerprints, the algorithm first computes the AOV score (a distance) of all AOVs in two fingerprints. Then a preliminary matching algorithm adds a pair of AOV into the matched pair set if their AOV score is within a threshold. If the number of matched pairs is more than a threshold after preliminary matching is done, a fine matching algorithm is performed. Fine matching is able to process deformed minutiae and adds new matched AOV pairs into the matched pair set. Finally AOV scores of all matched pairs are summed. If the overall score is more than a threshold, the two fingerprints are matched. Refer to [24] for an example implementation of the AOV-based approach.

- We found that the correlation-filter based method performs well consistently for both whole-finger and fingertip fingerprints. It works in our case as follows: Fourier transform is applied to each input image and captures local ridge structure, ridge frequency, and global ridge pattern. A correlation filter is the Fourier transform of a template, which is a training image of the finger of interest. A person's finger is enrolled by this correlation filter. In the verification stage, Fourier transform of a test image is derived and multiplied by the correlation filter corresponding to the finger of the person of interest. When the test image and template are matched, a large inner product is expected. Refer to [25] for an example implementation of the correlation-filter-based approach.

4. EVALUATION

We have implemented the fingerprint attack against touch-enabled devices including iPad 2, iPhone 4s and Android Phone (HTC G7). In this section, we use real-world experiments to demonstrate the feasibility and effectiveness of the fingerprint attack.

4.1 Experiment Setup

We performed experiments on iPad 2, iPhone 4s and Android Phone (HTC G7) to infer the screen unlock PIN/password/pattern and passwords for applications such as *Apple Store*. In Apple IOS, a user inputs a 4-digits *screen unlock pin* (denoted as SUP) on a 12-button keypad. Android has the *PIN mode* (denoted as PINM), *password mode* (denoted as PM). A 12-buttons keypad is used in the PIN mode, while a 26-letter keyboard is used in the password mode. A password needs to be at least four characters in Android. The *application password* (denoted as AP) is input on a 26-letters keyboard in both Android and IOS and a user can switch the keyboard between the letter panel and number panel. In our experiments, we use the letter panel to input the password and will explore issues caused by the panel switch in our future work.

Variable password length introduces complication in inferring passwords from fingerprint images. We address two cases. In the case of fixed-length password, we assume the attacker knows the password length. For instance, the screen unlock pin length in IOS is four. Based on the number of repeating characters, we classify the 4-digits IOS screen unlock pin into five patterns: 1-1-1-1, 2-1-1, 2-2, 3-1, and 4-0. The number in each pattern is the repeating times of a pin digit. Therefore, pattern 1-1-1-1 refers to a pin of 4 different digits and 4-0 is a pin with 4 repeating digits. In the second case, we assume that the attacker does not know the password length and all password characters are different. The attacker has to infer the password length to derive the password. We leave the discussion of inferring variable-length passwords with repeating keys as our future work.

4.2 Detection Rate

Our experiments show that the fingerprint attack can work effectively and efficiently. We performed 130 groups of experiments to infer the IOS unlock pin from fingerprint images. On iPad 2 and iPhone 4s, we conducted 30 groups of experiments for 1-1-1-1, 2-1-1, 2-2, 3-1 and 4-0 pins respectively. Figure 16 shows that detection rate for iPad 2 and iPhone 4s is as high as 68.5% and 63.3% respectively.

For the *PIN mode* and *password mode* in Android, we set password length as 4, 5, and 6, and conducted 10 groups of experiments for each length. 2 groups of experiments for each length contain repeating characters. Note that a 12-button keypad is used in the *PIN mode* and a 26-letter keyboard is used in the *password mode*. Figure 16 shows that detection rate in the *PIN mode* and *password mode* for a fixed length password (4, 5, and 6 characters) is as high as 60.0% and 56.6% respectively. We also conducted 30 groups of experiments for inferring passwords without repeating characters. Detection rates for the *PIN mode* and *password mode* can reach 70.0% and 64.3% respectively.

For *application password*, we use iPad 2, take Apple Store as an example and conducted 90 groups of experiments. An Apple Store account requires a password of at least 8 characters. The lifetime of a session in Apple Store is 15 minutes due to default settings. After timeout, the user has to log in again with the given account name. We set password length as 8, 9, and 10, and performed 30 groups of experiments for each length. 6 groups of experiments for each length contain repeating characters. Figure 16 shows that detection rate for fixed-length passwords with repeating characters on iPad 2 is 47.8%. After removing the 6 groups of passwords with repeating keys, detection rate for passwords without repeating characters on iPad 2 is 52.8%.

Figure 17 compares detection rate for pins of five patterns, derived from 130 groups of experiments on iPad 2 for inferring the screen unlock pin. It can be observed that with the threshold-based strategy in Section 3.4.2, detection rate can reach more than 70.0%. It demonstrates that this strategy can deal with overlapped fingerprints effectively.

Figure 18 shows detection rate in terms of password length. We used 10 groups of experiments for each length in Android in the *password mode*. Detection rate can reach more than 50%. We can also observe that detection rate is not sensitive to password length in the fingerprint attack.

Figure 19 illustrates that the detection rate of pressed key by K-means is far lower than detection rate by the threshold based approach. This observation verifies our analysis in Section 3.4.2.

4.3 Differentiating Fingerprints from Multiple Persons Sharing a Device

Recall that Filter-bank approach needs the local points and get the whole structure and it is not effective to verify the fingertips. Adjacent orientation vector system has a poor performance when there exists some dominant fingerprints with rich minutiae. As dominate fingerprints have more similar minutia pairs when verified with other fingers, it is highly possible that the other fingertips will be recognized as the dominate fingerprints. In comparison, correlation-filter based method does not require local core points to get the global structure and it can also get the local structure.

We adopted the correlation-filter approach for fingerprint comparison and conducted two groups of experiments to evaluate performance of differentiating fingerprints from multiple persons sharing a device. In the first group of experiments, we assume training fingerprints from people sharing a device are available. This assumption refers to scenarios where an attacker knows victims and collects training fingerprint images before-hand from objects touched by victims. We collected fingerprint images from 10 fingers of 6 people. Each finger is used to input a 4-digits pin on iPad 2 and produces 4 fingerprint (sub-)images. For each person, we choose one fingerprint image from each finger and use the 10 fingerprint images for 10 fingers as a training set. We choose one person as the target. The remaining 120 images are divided into 6 test sets with 3 fingerprint images for each finger of a person. *True positive rate* is defined as the probability that the fingerprint of the target is identified as one belonging to the target. *False positive rate* is defined as the probability that the testing image, which doesn't belong to the owner, is identified as one belonging to the target.

Figure 20 shows the true positive rate and false positive rate for the first group of experiments. The correlation fil-

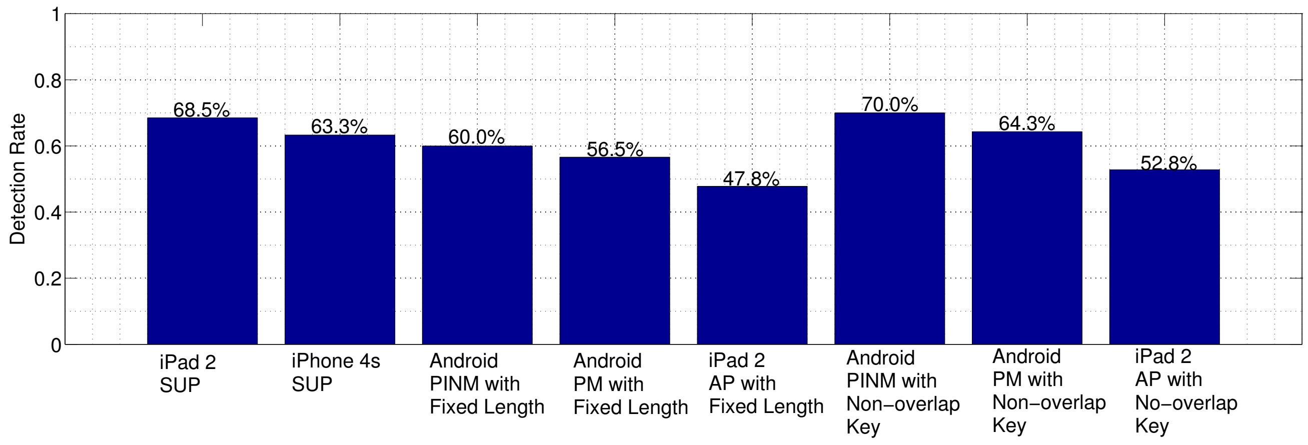


Figure 16: Detection Rate for iPad, iPhone 4s and Android Phone

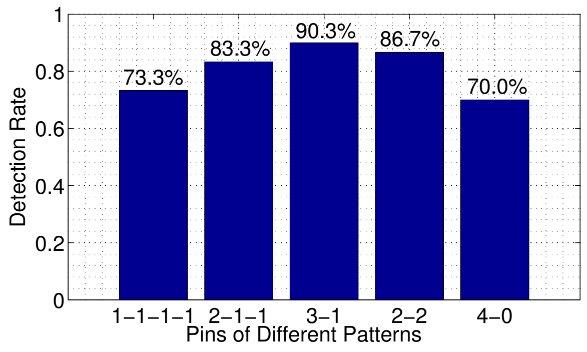


Figure 17: Detection Rate for IOS Pins of Different Patterns

ter performs better with the whole-finger images than with finger-tip images. The false positive rate for finger-tips is very high and can reach more than 40% in the case of differentiating two people. This confirms our analysis: whole fingers have distinguishable global structures and richer local structures and are easier to identify.

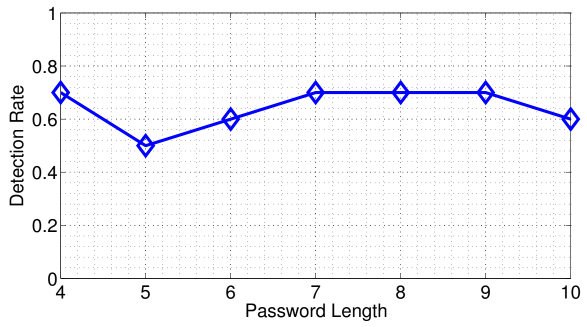


Figure 18: Password Length v.s. Detection Rate

In the second group of experiments, we want to test if fingerprints on a touch screen can be grouped together without training sets. There are 6 test sets from 2 fingers (index and middle fingers) of 3 persons, and 4 fingerprint images are

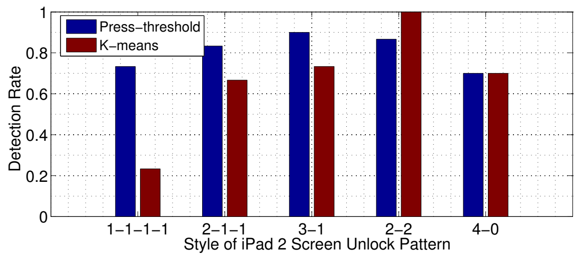


Figure 19: Detection Rate by K-means Clustering

Table 1: Differentiating Fingerprints from Multiple Persons *Without* Training Fingerprints

	Whole Finger	Finger Tip
True Positive	69.44%	44.44%
False Positive	0.63%	0%

from one finger in each test set. We verify whether two fingerprint images are from the same finger by peer comparison. Table 1 shows the true positive rate and false positive rate. Tests with whole fingers has much higher true positive rate than tests with finger tips since whole fingers have more significant global structure information and richer local details than fingertips. Tests with both whole fingers and finger tips produce low false positive rate. Therefore, if this method verifies two fingerprint images match, it is highly possible that these two fingerprint are from the same finger. Therefore, we can identify the passwords tapped by *this finger* on the touch screen.

5. DISCUSSION - COUNTERMEASURE

In this section, we briefly discuss possible countermeasures protecting touch-enabled devices against the fingerprint attack.

To defend against fingerprint attacks, one can ensure the physical security of the touch enabled devices and the environment where such devices are deployed. Surveillance cameras can be deployed to ensure that malicious behavior is captured. We may also make photographing fingerprints

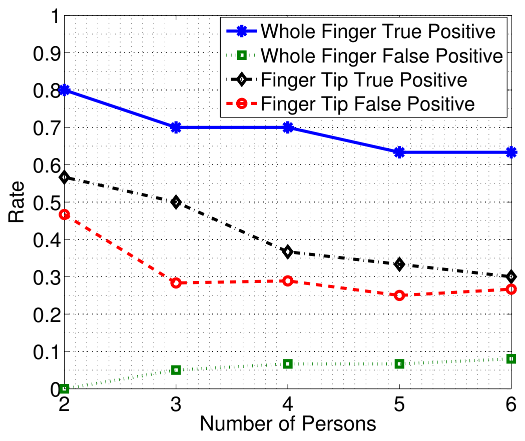


Figure 20: Differentiating Fingerprints from Multiple Persons *With* Trained Fingerprints

on touch screen hard. People have suggested that abrasive membrane is able to reduce the smudge of fingerprints on the touch-enabled devices. However, ridges in human skin left on the surface are still visible after dusting strategy in our experiments, and thus it is ineffective to use an abrasive membrane to defend against attacks. Of course, one obvious approach to remove fingerprints on touch screen is to clear the screen with appropriate cloth after each use. The approach may not be very convenient in many scenarios. A special glass coating technique such as [20] may also be used to prevent fingerprint on the surface. The effectiveness of this anti-fingerprint coating needs further evaluation and we will explore this technique in our future work.

One effective countermeasure against fingerprint attacks is to randomize the software keyboard on touch-enabled devices. A randomized keyboard may not be convenient in many applications. We suggest that at least the software keyboard for inputting sensitive information including passwords and pin numbers should be randomized in an appropriate way. A randomized keyboard can be implemented in two fashions: the operating system implements it or an application itself implements one. We observed very few applications today implement a randomized software keyboard on smartphones.

If a randomized keyboard cannot be used, here are some compromised solutions. To Mac IOS, we suggest the system increase the length of the unlocking screen password which is four now and apply safer screen unlock strategies. To Android, in the PIN mode, we suggest users to contain overlap numeric keys in the password because non-fixed length password with overlapping numeric keys is able to definitely and sharply drop the detection rate of the fingerprint attack. In the password mode, we suggest users to include both numbers and letters in the password because numbers and letters in the password mode are from different panels switched by the shift key so that the attacker can not distinguish the panels from which fingerprints on the surface are left.

6. CONCLUSION

This paper investigates fingerprint attacks against touch-enabled devices to infer user pins or passwords from fingerprints left on a touch screen. In the fingerprint attack, an attacker dusts the touch screen and reveal hidden fingerprints. In our research, iPhone is used for photographing the fingerprint and keypad. Various image processing techniques are used to process the two images, sharpening fingerprints and removing the background in order to automatically map the fingerprints to specific keys and recover password characters. We performed extensive experiments and the results show that the fingerprint attack works effectively and efficiently. We also briefly discussed countermeasures to the fingerprint attack and suggest that a randomized software keyboard is a feasible solution while most touch-enabled devices such as smartphones have not implemented this functionality.

7. ACKNOWLEDGEMENTS

We would like to thank Yinjie Chen, Zhongli Liu, Xian Pan, Junwei Huang, Dan Zhao, Shan Lu, Peng Xia and Yang Zhang for the considerable amount of time spent providing us their fingerprints. This work was supported in part by National Key Basic Research program of China under grants 2010CB328104, National Natural Science Foundation of China under grants 60903162 and 60903161, US NSF grants 1116644, 0942113, 0958477 and 0943479, China National Science and Technology Major Project under grants 2010ZX01044-001-001, China Specialized Research Fund for the Doctoral Program of Higher Education under grants 20110092130002, Jiangsu Provincial Natural Science Foundation of China under grants BK2008030, Jiangsu Provincial Key Laboratory of Network and Information Security under grants BM2003201, and Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under grants 93K-9. Any opinions, findings, conclusions, and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

8. REFERENCES

[1] D. Asonovand and R. Agrawal. Keyboard acoustic emanations. In *Proceedings of IEEE Symposium on Security and Privacy*, 2004.

[2] A. J. Aviv, K. Gibson, E. Mossop, M. Blaze, and J. M. Smith. Smudge attacks on smartphone touch screens. In *Proceedings of Workshop on Offensive Technology WOOT*, 2010.

[3] S. L. Bacharach, M. V. Green, D. Vitale, G. White, M. A. Douglas, R. . Bonow, and S. M. Larson. Optimum Fourier Filtering of Cardiac Data: A Minimum-Error Method: Concise Communication. *THE JOURNAL OF NUCLEAR MEDICINE*, 24:1176–1184, 1984.

[4] D. Balzarotti, M. Cova, and G. Vigna. Clearshot: Eavesdropping on keyboard input from video. In *Proceedings of the 29th IEEE Symposium on Security and Privacy (S&P)*, 2008.

[5] M. Becher, F. C. Freiling, J. Hoffmann, T. Holz, S. Uellenbeck, and C. Wolf. Mobile security catching up? revealing the nuts and bolts of the security of mobile devices. In *Proceedings of IEEE Symposium on Security and Privacy*, 2011.

[6] L. Cai and H. Chen. TouchLogger: Inferring keystrokes on touch screen from smartphone motion.

In *Proceedings of the 6th USENIX Workshop on Hot Topics in Security (HotSec)*, 2011.

[7] D. Couzin. Depths of Field. *SMPTE Journal*, 1982.

[8] Crime Scene. Latent fingerprint kit, professional. `http://www.crimescene.com/store/index.php?main_page=product_info&products_id=103`, 2012.

[9] EVIDENT, INC. Forensic field kits. `http://www.evidentcrimescene.com/cata/kits/kits.html`, 2012.

[10] A. P. Felt, M. Finifter, E. Chin, S. Hanna, and D. Wagner. A survey of mobile malware in the wild. In *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices (SPSM)*, 2011.

[11] N. G.S., X. Tong, X. Tang, and D. Shi. Adjacent Orientation Vector based Fingerprint Minutiae Matching System. In *Proceedings of the 17th International Conference on Pattern Recognition*, August 2004.

[12] A. K. Jain, S. Prabhakar, L. Hong, and S. Pankanti. Filterbank-Based Fingerprint Matching. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 9(5), May 2000.

[13] D. F. KUNE and Y. KIM. Timing attacks on pin input devices. In *Proceedings of the 17th ACM CCS*, 2010.

[14] D. Lee. The state of the touch-screen panel market in 2011. `http://www.walkermobile.com/March_2011_ID_State_of_the_Touch_Screen_Market.pdf`, March 2011.

[15] F. Maggi, A. Volpatto, S. Gasparini, G. Boracchi, and S. Zanero. A fast eavesdropping attack against touchscreens. In *Proceedings of the 7th International Conference Information Assurance and Security (IAS)*, 2011.

[16] L. Mandelstam. Light Scattering by Inhomogeneous Media. *Zh. Russ. Fiz-Khim. Ova.*, 58(381), 1926.

[17] P. Marquardt, A. Verma, H. Carter, and P. Traynor:. (sp)iPhone: decoding vibrations from nearby keyboards using mobile phone accelerometers. In *Proceedings of ACM CCS*, 2011.

[18] J. P. Mock. *Basic Latent Print Development.* Lighting Powder Company, Inc., 1993.

[19] K. Mowery, S. Meiklejohn, and S. Savage. Heat of the moment: Characterizing the efficacy of thermal camera-based attacks. In *Proceedings of Workshop On Offensive Technologies (WOOT)*, 2011.

[20] Nanogate. Fine material with a permanently fine appearance. `http://www.nanogate.de/en/product-enhancement/functions/nanotension/anti-fingerprint.php`, 2012.

[21] N. Otsu. A Threshold Selection Method from Gray-level Histograms. *IEEE Trans. Sys., Man., Cyber*, 9(1):62–66, 1979.

[22] E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang. Accessory: Keystroke inference using accelerometers on smartphones. In *Proceedings of The Thirteenth Workshop on Mobile Computing Systems and Applications (HotMobile).* ACM, February 2012.

[23] R. Raguram, A. White, D. Goswami, F. Monrose, and J.-M. Frahm. iSpy: Automatic reconstruction of typed input from compromising reflections. In *Proceedings of ACM CCS*, 2011.

[24] L. Rosa. Adjacent orientation vector based fingerprint minutiae matching system. `http://www.advancedsourcecode.com/aovminutiae.asp`, 2012.

[25] L. Rosa. Correlation-filter based method. `http://www.advancedsourcecode.com/fingerprint4.asp`, 2012.

[26] L. Rosa. Filterbank-based fingerprint matching. `http://www.advancedsourcecode.com/fingerprint.asp`, 2012.

[27] L. J. van Vliet and I. T. Young. A Nonlinear Laplace Operator as Edge Detector in Noisy Images. *Computer Vision, Graphics, and Image Processing*, 45(2):167–195, February 1989.

[28] K. Venkataramani and B. V. K. V. Kumar. Performance of composite correlation filters in fingerprint verification. *Optical Engineering*, 43(8), August 2004.

[29] M. Vuagnoux and S. Pasini. Compromising electromagnetic emanations of wired and wireless keyboards. In *Proceedings of the 18th conference on USENIX security symposium*, 2009.

[30] Z. Xu, K. Bai, and S. Zhu. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In *Proceedings of The ACM Conference on Wireless Network Security (WiSec)*, 2012.

[31] M. Zalewski. Cracking safes with thermal imaging. `http://lcamtuf.coredump.cx/tsafe/`, 2005.

[32] L. Zhuang, F. Zhou, and J. D. Tygar. Keyboard acoustic emanations revisited. In *Proceedings of the 12th ACM CCS*, 2005.