May 2020

Latent Fingerprint Detection Using Rotationally-Invariant Vectors

Firas Sammoura

Jean-Marie Bussat

Latent Fingerprint Detection Using Rotationally-Invariant Vectors

Abstract:

This publication describes techniques and methods implemented on a computing device, directed at detecting latent fingerprint reactivation attacks (*e.g.*, the exploitation of latent fingerprints deposited on fingerprint sensors to gain access to computing devices). In aspects, for each fingerprint authentication event, an on-device fingerprint-matching algorithm extracts small patches from a fingerprint image (a "Verify Image") and transforms the patches into rotationally-invariant vectors. The fingerprint-matching algorithm then calculates the similarity between the vectors of the Verify Image and vectors of a stored, authentic fingerprint (an "Enrolled Image"). If the computing device validates the fingerprint (*e.g.*, authorizes the fingerprint and determines it is not a latent fingerprint by the steps described below), the computing device permits the user access. The fingerprint image, formerly Verify Image, is now a "Previous Image." Finally, the fingerprint-matching algorithm merges the vectors from the Enrolled Image and Previous Image using a rotation and translation matrix (a "Previous Matrix"). The computing device temporarily stores the Previous Matrix.

Upon a successive fingerprint authentication event, the algorithm extracts small patches from a new Verify Image, transforms the patches into rotationally-invariant vectors, calculates the rotation and translation matrix relative to the Enrolled Image (a "Verify Matrix"), and computes the similarity between the Previous Matrix and the Verify Matrix. If the rotation and translation matrices are identical, then the computing device can reject the latent fingerprint.

Keywords:

Optical fingerprint sensor, biometric recognition systems, biometric authentication, fingerprint matching, authorized access, fingerprint residual, device unlock, latent fingerprints, reactivation attacks, fingerprint-matching algorithms, NxN blocks, rotation and translation, rotationally-invariant vectors

Background:

Biometric recognition systems use unique biological characteristics (*e.g.*, voice, retina, fingerprint, and the like) to reliably identify and authenticate users. Of all the available biometric recognition systems, fingerprint scanning using optical fingerprint sensors is the most prevalent due to its low cost, high efficiency, and relative accuracy. An optical fingerprint sensor can be directly integrated under a display screen of a computing device, and thereby afford seamless authentication. Given these advantages, computing device users often rely on fingerprint scanning to secure and access their computing devices. As with all systems, however, optical fingerprint sensors can be tricked, fooled, or spoofed. One such approach to fool fingerprint scanners involves the use of latent fingerprints. This approach, known as a latent fingerprint reactivation attack, attempts to reactivate a computing device through the utilization of a latent fingerprint deposited above an optical fingerprint sensor. In short, latent fingerprint reactivation attacks involve unauthorized users exploiting authorized, latent fingerprints deposited above optical fingerprint sensors to gain access to computing devices. Figure 1, below, illustrates a latent fingerprint deposited on a computing device directly above an optical fingerprint sensor.

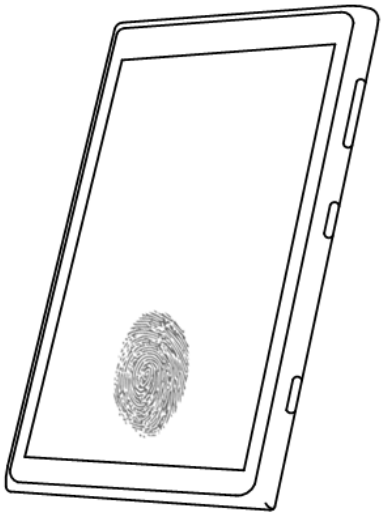

Figure 1

As illustrated, the depicted computing device is a smartphone. Still, other computing devices, for example, tablets or watches that utilize optical fingerprint sensors for authentication purposes, may also be vulnerable to latent fingerprint reactivation attacks. The smartphone illustrated contains a latent fingerprint deposited directly above an optical fingerprint sensor that resides beneath the display cover glass. As a result, an attacker may attempt to reactivate the computing device using this latent fingerprint.

One approach to thwart reactivation attacks involves the prohibition of identical, consecutive fingerprint samples. While this solution works well for small-area optical fingerprint sensors (small-area FPS), large-area optical fingerprint sensors (large-area FPS) may fail to identify identical, consecutive fingerprint samples. Figures 2A and 2B, below, illustrate two computing devices: one with a small-area FPS and another with a large-area FPS.

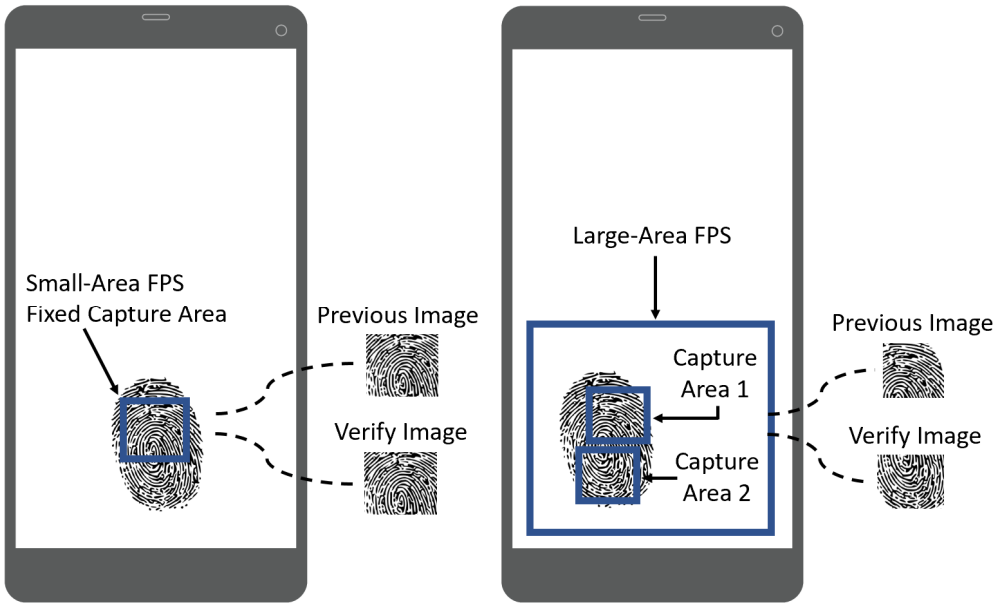

Figure 2A **Figure 2B**

Figure 2A illustrates a smartphone with a small-area FPS, while Figure 2B illustrates a smartphone with a large-area FPS. In both cases, the users authenticated themselves to their smartphone using fingerprint scanning. Both optical fingerprint sensors captured an image of a live fingerprint (Previous Image). In using fingerprint scanning, the users left a latent fingerprint on the display cover glass. Sometime later, an attacker attempts latent fingerprint reactivation attacks on both smartphones. During these attacks, both optical fingerprint sensors captured an image of the latent fingerprint (Verify Image).

Smartphones with a small-area FPS have a fixed capture area; hence, successively reusing the latent fingerprint results in identical fingerprint images, as illustrated in Figure 2A. The optical fingerprint sensor captured identical, consecutive fingerprint images (Previous Image and Verify Image), and as a result, the smartphone could detect the latent fingerprint reactivation attack. Smartphones with a large-area FPS, on the other hand, have a dynamic capture area. In other words, the capture area can shift vertically and/or horizontally, depending on where the peak touch

signal is detected. Hence, reusing a latent fingerprint may result in two different fingerprint images, as is illustrated in Figure 2B, where the Previous Image is different from the Verify Image. As a result, the smartphone in Figure 2B did not identify a latent fingerprint reactivation attack.

Since many users rely on large-area optical fingerprint sensors to secure and access their computing devices, latent fingerprint reactivation attacks pose serious security and privacy concerns. Therefore, it is desirable to safeguard computing devices from latent fingerprint reactivation attacks. To this end, by generating rotation and translation matrices of a Verify Image and a Previous Image both relative to an Enrolled Image, comparing the matrices, computing devices can quickly identify and reject latent fingerprints if the rotation and translation matrices are identical.

Description:

This publication describes techniques and methods implemented on a computing device, directed at detecting latent fingerprint reactivation attacks. For each fingerprint authentication event, an optical fingerprint sensor, in a first step, captures a fingerprint image (a "Verify Image"). All Verify Images reference a single, 2-dimensional coordinate system. Figure 3 illustrates the referenced coordinate system.

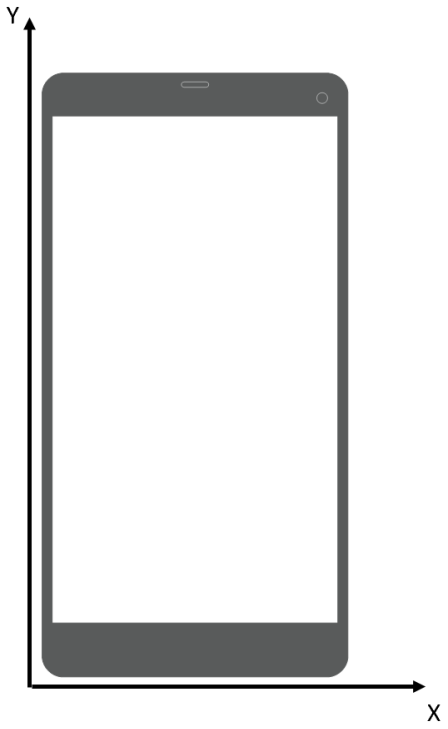

Figure 3

In the next step, a fingerprint-matching algorithm divides each Verify Image into "N" number of patches with a sliding distance of one pixel. The fingerprint-matching algorithm then extracts the patches and calculates rotationally-invariant vectors for each of the patches in reference to the established coordinate system.

The fingerprint-matching algorithm extracts rotationally-invariant vectors from each patch by including the following:

- Rotationally invariant Absolute-value Fast Fourier Transforms (AFFTs) of each block;
- The patches' x-position and y-position — the Cartesian coordinates;
- The patches' polar representation of the Cartesian coordinates; and
- The patches' Fast Fourier Transforms (FFTs) of the polar representation.

The fingerprint-matching algorithm then calculates the similarity between the vectors of the Verify Image and vectors of a stored, authentic fingerprint (an "Enrolled Image"). The fingerprint-matching algorithm also outlines the x-coordinate, the y-coordinate, and the angle correspondence between the Verify Image blocks and their matching blocks in the Enrolled Image. Moreover, the algorithm calculates the translation in the x-direction and y-direction for each Verify Image block. Finally, the fingerprint-matching algorithm merges the vectors from the Verify Image and the Enrolled Image using a rotation and translation matrix.

If the computing device validates the fingerprint, it permits the user access and temporarily stores the rotation and translation matrix (a "Previous Matrix") of the Verify Image, now referred to as a Previous Image. The Previous Matrix may be temporarily stored on-device until replaced by a successive rotation and translation matrix of a new Previous Image. Further, a user may be provided with controls allowing the user to make an election as to both if and when systems, programs, and/or features described herein may enable the collection and storage of a Previous Matrix. The computing device can be configured to only use the information after the computing device receives explicit permission from the user of the computing device to use the data.

Upon a successive fingerprint authentication event, the algorithm again extracts small patches from a Verify Image, transforms the patches into rotationally-invariant vectors, calculates the similarity between the vectors of the Verify Image and the vectors of the Enrolled Image, and merges the Verify Image vectors and the Enrolled Image vectors using a rotation and translation matrix (a "Verify Matrix"). The algorithm then compares the Verify Matrix to the Previous Matrix. If the matrices are identical, then a latent fingerprint reactivation attack can be identified.

Take, for instance, a user (John) and his smartphone. John authenticates himself to his smartphone using fingerprint scanning and, in so doing, deposits a latent fingerprint on the display

cover glass. John's smartphone authenticates his fingerprint by means of a fingerprint-matching algorithm, which both validates his fingerprint (*e.g.*, verifies the fingerprint matches the enrolled fingerprint) and confirms that his fingerprint is not a latent fingerprint. Finally, the fingerprint-matching algorithm temporarily stores the rotation and translation matrix of his fingerprint image. John, soon after, turns off his smartphone, sets it down, and walks away. An unauthorized user then grabs the phone and attempts a latent fingerprint reactivation attack. Figure 4 illustrates John's smartphone and the latent fingerprint.

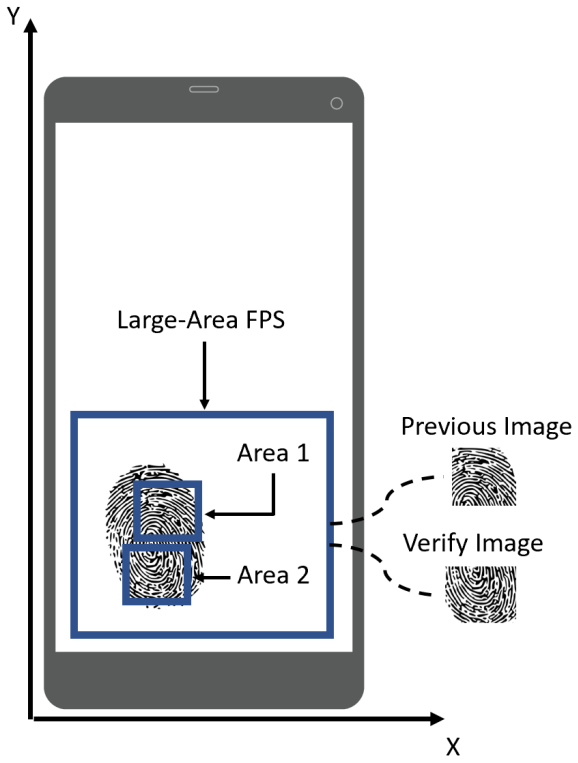

Figure 4

As illustrated, the smartphone contains a large-area fingerprint sensor. The first capture area (Area 1) was the capture area where the computing device first captured an image of John's fingerprint (Previous Image). The fingerprint-matching algorithm calculated a Previous Matrix for the Previous Image relative to an Enrolled Image. The smartphone stored this Previous Matrix.

The second capture area (Area 2) was the capture area where the computing device captured the newest fingerprint image (Verify Image). The smartphone captured the Verify Image during an authentication event initiated by an unauthorized user attempting a latent fingerprint reactivation attack. In response to the authentication event, the fingerprint-matching algorithm calculates a Verify Matrix for the Verify Image relative to the Enrolled Image and compares it to the Previous Matrix. The fingerprint-matching algorithm identifies identical rotation and translation matrices, and as a result, rejects the attacker access.

Advantages of such an approach to identify latent fingerprint reactivation attacks include identifying latent fingerprints with minimal power and latency since, for example, latent fingerprint detection can operate simultaneously to fingerprint validation events. Additionally, the algorithm only needs to calculate a few rotationally-invariant vectors to determine the similitude of the rotation matrices before rejecting the latent fingerprint. Finally, the fingerprint-matching algorithm can compare Verify Images to more than one Previous Image in an effort to ensure no latent fingerprint enables access to the device, regardless of when the latent fingerprint was deposited on the display glass.

In addition to the above descriptions, during a no-intent authentication event, the fingerprint-matching algorithm can operate in low-power mode to capture fingerprint images, extract blocks, transform the blocks to rotationally-invariant vectors, calculate a Previous Matrix for the fingerprint image relative to the Enrolled Image, and temporarily store the rotation and translation matrix. A no-intent authentication event may occur when users handle their device with no intent to access their device and yet deposit latent fingerprints on the display glass. For example, when a user grabs their phone off a table or from their pocket. Therefore, regardless of

when or how a latent fingerprint was deposited on the display cover glass, the fingerprint-matching algorithm can protect the computing device from latent fingerprint reactivation attacks.

Further still, an anti-latent detection algorithm (*e.g.*, a machine-learned model) can be employed to identify latent fingerprints. Employing this anti-latent detection algorithm in combination with the fingerprint-matching algorithm provides a two-level security scheme by which latent fingerprints can be identified and rejected.

In summary, this publication describes techniques and methods implemented on a computing device, directed at detecting latent fingerprint reactivation attacks. In aspects, for each fingerprint authentication event, an on-device fingerprint-matching algorithm extracts small patches from a Verify Image and transforms the patches into rotationally-invariant vectors. The fingerprint-matching algorithm then calculates the similarity between the vectors of the Verify Image and vectors of the Enrolled Image. If the computing device validates the fingerprint, then the computing device permits access to the user. The fingerprint image, formerly Verify Image, is now a Previous Image. Finally, the fingerprint-matching algorithm merges the vectors from the Enrolled Image and Previous Image, generating a Previous Matrix. The computing device temporarily stores the Previous Matrix.

Upon a successive fingerprint authentication event, the algorithm extracts small patches from a new Verify Image, transforms the patches into rotationally-invariant vectors, calculates a Verify Matrix relative to the Enrolled Image, and computes the similarity between the Previous Matrix and the Verify Matrix. If the rotation and translation matrices are identical, then the computing device can reject the latent fingerprint.

References:

[1] Patent Publication: EP 1993062 A2. Method and biometric system for recognising latency impressions. Priority Date: May 15, 2007.

[2] Patent Publication: US 20180330147 A1. Method and apparatus for recognizing fingerprint. Priority Date: April 15, 2015.

[3] Patent Publication: US 20190026535 A1. Processing method and electronic device. Priority Date: February 24, 2016.