InfinityGauntlet: Expose Smartphone Fingerprint Authentication to Brute-force Attack

Yu Chen and Yang Yu, *Xuanwu Lab, Tencent;* Lidong Zhai, *Institute of Information Engineering, Chinese Academy of Sciences*

https://www.usenix.org/conference/usenixsecurity23/presentation/chen-yu

This paper is included in the Proceedings of the 32nd USENIX Security Symposium.

August 9–11, 2023 • Anaheim, CA, USA

978-1-939133-37-3

InfinityGauntlet: Brute-force Attack on Smartphone Fingerprint Authentication

Yu Chen
Xuanwu Lab, Tencent

Yang Yu*
Xuanwu Lab, Tencent

Lidong Zhai
Institute of Information Engineering, Chinese Academy of Sciences, China

Abstract

Billions of smartphone fingerprint authentications (SFA) occur daily for unlocking, privacy and payment. Existing threats to SFA include presentation attacks (PA) and some case-by-case vulnerabilities. The former need to know the victim's fingerprint information (e.g., latent fingerprints) and can be mitigated by liveness detection and security policies. The latter require additional conditions (e.g., third-party screen protector, root permission) and are only exploitable for individual smartphone models.

In this paper, we conduct the *first* investigation on the general zero-knowledge attack towards SFA where no knowledge about the victim is needed. We propose a novelty fingerprint brute-force attack on off-the-shelf smartphones, named INFINITYGAUNTLET. Firstly, we discover design vulnerabilities in SFA systems across various manufacturers, operating systems, and fingerprint types to achieve unlimited authentication attempts. Then, we use SPI MITM to bypass liveness detection and make automatic attempts. Finally, we customize a synthetic fingerprint generator to get a valid brute-force fingerprint dictionary.

We design and implement low-cost equipment to launch INFINITYGAUNTLET. A proof-of-concept case study demonstrates that INFINITYGAUNTLET can brute-force attack successfully in less than an hour without any knowledge of the victim. Additionally, empirical analysis on representative smartphones shows the scalability of our work.

1 Introduction

Smartphone fingerprint authentication (SFA) has become the most preferred biometric used on smartphones due to its adequate security and comfortable user experience [28]. As a result, SFA is applied to various applications, including screen unlocking, privacy application login, and payment authentication. Unfortunately, unlike fingerprint authentication on other devices (e.g., fingerprint attendance machine), neither the algorithm nor the testing tool of SFA is publicly available, resulting in insufficient research on its security.

*The corresponding author.

Presentation attack (PA) [29] has long been considered the main threat to SFA. The classic steps of PA include: 1) capture a latent fingerprint of the victim from a flat surface (e.g., phone screen); 2) process and print the fingerprint on an artificial material(e.g., silica gel); 3) present the artifact to the fingerprint sensor. However, the attack effect of PA is limited for the following reasons. Firstly, adversaries must collect at least one clear latent fingerprint of the victim, which is difficult to satisfy in practical scenarios [37], let alone most smartphones have an oleophobic coating on the screen to prevent fingerprint residual. Secondly, the deployment of liveness detection and quality check make artifacts easier to be identified. Last, the limit on the maximum number of failed attempts (hereinafter referred to as "attempt limit") makes adversaries have to attack successfully within a limited number of attempts.

In addition to PA, there have been some case-by-case vulnerabilities in recent years, which can be divided into two categories. (1) Fingerprint template update unwarily [5,8,12]: for example, Samsung Galaxy S10 and Note10 allow unlocking operations via arbitrary fingerprints in certain situations involving a third-party screen protector. However, these vulnerabilities are only accidentally exploitable when the textures of the screen protector trigger unnecessary template updates. (2) Implementation bug in SFA systems [6,7]: for example, OnePlus 7 Pro contains leftover debug code that allows a root user to obtain fingerprint images. However, these vulnerabilities require adversaries to know the PIN/password and get privileged permissions. Furthermore, they are only exploitable for individual phone models, in other words, they are not generic.

To the best of our knowledge, none has investigated the general SFA attack that does not require victim information. Inspired by user feedback [10,14,15] of unexpected fingerprint unlocking caused by SFA's false acceptance rate (*FAR*), we study the feasibility of fingerprint brute-force attacks to achieve the general zero-knowledge attack. In cryptography, a brute-force attack consists of an attacker submitting many passwords or passphrases until eventually guessing correctly. However, the following challenges exist when migrating the brute-force attack concept from cryptography to SFA.

(a) *How to bypass attempt limit to achieve unlimited authentication attempts?* We discover two general types of vulnerabilities in SFA systems: Cancel-After-Match-Fail (CAMF) and Match-After-Lock (MAL), either of which can be exploited to bypass the attempt limit. Instead of implementation bugs, CAMF and MAL leverage design faults in the biometric authentication framework. So they exist across various manufacturers, operating systems, and fingerprint types.

(b) *How to bypass liveness detection and make automatic attempts?* The research in [23] shows that adversaries can easily bypass liveness detection if they can inject images at the hardware layer. Inspired by it, we find the insufficient protection of fingerprint data on the serial peripheral interface (SPI) and thus propose an approach to make man-in-the-middle (MITM) attacks for fingerprint image hijacking. Further, we design an auto-clicker to trigger fingerprint authentication automatically.

(c) *How to get a large number of fingerprints for attempts?* Tens of thousands of fingerprints are required to perform brute-force attacks due to the low *FAR* of the matching algorithm. A straightforward way to obtain them is to recruit volunteers to collect real fingerprints. Unfortunately, it is expensive and tedious for both the data collection technicians and the subjects providing the data. Besides, it is difficult to share with others due to privacy legislation that protects personal data. We propose a synthetic fingerprint generation method customized from SFinGe [22] to generate infinite synthetic fingerprints for free.

After addressing the above challenges, we successfully adapt the brute-force attack to SFA. **We name this new threat model INFINITYGAUNTLET, where INFINITY and GAUNTLET represent its two core techniques: attempt limit bypassing and fingerprint image hijacking, respectively**.

We design and implement low-cost proof-of-concept equipment that will be open-source to the academic community. Using the equipment, we show a case study of INFINITYGAUNTLET on OnePlus 7 Pro. The experiment results of the case study prove that the attempt limit and liveness detection are bypassed completely, and the brute-force attack can succeed in less than an hour without any knowledge about the victim. Furthermore, we empirically analyse INFINITYGAUNTLET on other popular smartphones from top manufacturers, including Samsung, Huawei, Xiaomi, Oppo, Vivo, OnePlus, Honor, and Apple, which cover various operating systems (Android, HarmonyOS, and iOS) and fingerprint sensor types (optical, capacitive and ultrasonic). Results show that most of them are more or less affected by INFINITYGAUNTLET. We believe the affected list is by far not comprehensive. Nevertheless, it serves as a wake-up call to reconsider the design for preventing brute-force attacks.

In summary, we make the following major contributions:

- We first propose the general zero-knowledge fingerprint brute-force attack on off-the-shelf smartphones.
- We demonstrate the effectiveness of our approach through a case study and analyse its scalability on representative smartphones.
- We discover zero-day design vulnerabilities that can be exploited to bypass the attempt limit of SFA systems.
- We implement fingerprint hijacking through the hardware of SFA systems.
- We propose a synthetic fingerprint generation method for SFA.

2 Background

SFA technology is closed-source with few public materials. This section covers some basics of the state-of-the-art SFA system, especially security-related.

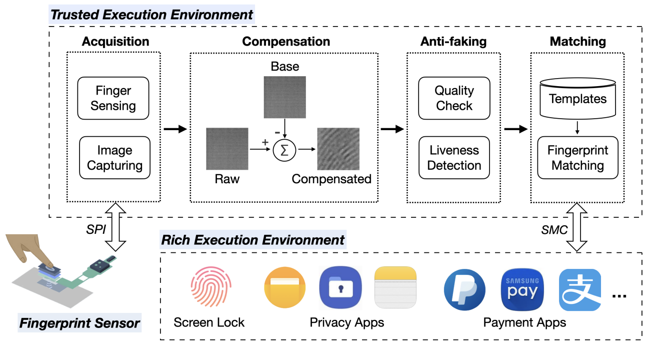


Figure 1: The authentication workflow of SFA.

2.1 Authentication Workflow

The fingerprint authentication process on smartphones consists of four main stages: *acquisition*, *compensation*, *anti-faking* and *matching*, as illustrated in Fig. 1.

Acquisition captures multiple fingerprint images when sensing a finger press and receives these raw images through SPI.

Compensation boosts the raw image quality by stacking a pre-captured *base image*. The *base image* contains fixed patterns mainly caused by screen pixels diffraction, vignetting effects, or fixed ground noise.

Anti-faking checks the quality of the compensated images and applies liveness detection to prevent PA. As fake fingers in PA are made with artificial materials, the liveness detection is based on materials features, including textures and ridge-valley contrast [39].

Matching measures the similarity between a query and enrolled fingerprints. Since the enrolled fingerprints are stored as irreversible templates (e.g., feature vectors) rather than images, the similarity calculation is done on fingerprint templates. Besides, instead of traditional minutiae-based methods [38], fingerprint matching on modern smartphones needs to ensure robustness in various environments. For example, they extract characteristic SIFT feature points for matching [35].

2.2 Security Policy

Besides the anti-faking, SFA systems incorporate many security policies [2]. The typical policies are as follows:
P1: Forbid SFA and challenge for primary authentication (e.g., PIN, pattern, password) if the number of failed attempts exceeds the attempt limit.
P2: Forbid SFA and challenge for the primary authentication once every 72 hours.
P3: Forbid SFA and challenge for the primary authentication after smartphone restart.
P4: Forbid SFA at least 30 seconds after five consecutive failed SFA attempts.
P5: Forbid SFA when the primary authentication is locked out temporarily.

In practice, we find that the attempt limit in **P1** varies from 5 to 50 depending on manufacturers and applications.

2.3 Architectural Security

An SFA architecture is considered secure if OS kernel compromises do not confer the ability to either read fingerprint images or inject synthetic images into the system to influence the authentication decision [13]. The trusted execution environment (TEE) is used to isolate the fingerprint sensor driver, fingerprint authentication logic, and fingerprint data in a secure environment.

Since Android 6, Google has enforced moving all fingerprint data manipulation into TEE, and most smartphone manufacturers adopt TrustZone-based TEE solutions. At nearly the same time, Apple uses another solution called Secure Enclave [1].

2.4 Security Metrics

Two metrics [13] are used to measure the security performance of an SFA system:

False acceptance rate (FAR) defines the metric of the chance that an SFA system mistakenly accepts a randomly chosen unenrolled fingerprint. According to the industry convention, an SFA system is considered secure enough if its *FAR* is lower than 0.002% [1, 9]. Measuring *FAR* requires collecting multiple fingerprint images from each subject and each finger. For example, a common practice is to set the minimum subject number, the finger number per subject, and the fingerprint image number per finger to 24, 4 (left thumb, left index, right thumb, right index) 50, respectively. This setting theoretically makes $(24\times4-1)\times50\times(24\times4)=456000$ negative claims for the *FAR* evaluation, and a secure SFA system should give positive confirmations 9 times at most.

Spoof acceptance rate (SAR) defines the metric of the chance that an SFA model accepts a fake fingerprint. This metric is first introduced by Google in Android 8.1 and categorizes SFA systems as either strong or weak security. An SFA system with a *SAR* of 7% or lower is at strong security, and anything above 7% is weak [4]. Measuring *SAR* uses 2D and 2.5D presentation attack instruments such as printed fingerprints or a molded replica [13].

3 Threat Model

3.1 Mathematical Derivation

A fingerprint sensor is defined by a mapping $\mathrm{S}:\mathcal{T}\to\mathcal{P}$, where $\mathcal{T}$ represents objects that can trigger the fingerprint sensor to capture images and $\mathcal{P}$ represent the images. The quality check module, liveness detection module, and fingerprint matching module described in Fig. 1 are defined by $\mathrm{Q}:\mathcal{P}\to\{0,1\}$, $\mathrm{L}:\mathcal{P}\to\{0,1\}$ and $\mathrm{R}:\langle\mathcal{T},\mathcal{T}\rangle\to\{0,1\}$, where "0" represents "Reject" and "1" represents "Accept".

We say that $\mathcal{V}=\{v_1...v_i...v_N\}\subset\mathcal{T}$ are N fingers enrolled by a victim. We can now define a fingerprint authentication process:

$$\mathrm{M}_{r=N}(\mathrm{S}(q)):=\mathrm{Q}(\mathrm{S}(q))\cdot\mathrm{L}(\mathrm{S}(q))\cdot\bigvee_{i=1}^{N}\mathrm{R}(\mathrm{S}(q),\mathrm{S}(v_i))$$

where q represents a query finger, and r represents the number of enrolled fingers.

Suppose an unenrolled finger $f\in\mathcal{T}-\mathcal{V}$ satisfy $\mathrm{Q}(\mathrm{S}(f))=1$ and $\mathrm{L}(\mathrm{S}(f))=1$, then *FAR* is equal to:

$$\begin{aligned} FAR :&= \Pr[\mathrm{M}_{r=1}(\mathrm{S}(f))=1] \\ &= \Pr[\mathrm{R}(\mathrm{S}(f),\mathrm{S}(v))=1] \end{aligned} \tag{1}$$

Since SFA systems allow enrolling multiple fingers (up to four or five fingers), another metric *false positive identification-error rate (FPIR)* [30] defines the metric of the chance that f are falsely matched by any element of $\mathcal{V}$:

$$\begin{aligned} FPIR :&= \Pr[\mathrm{M}_{r=N}(\mathrm{S}(f))=1] \\ &= \Pr\left[\bigvee_{i=1}^{N}\mathrm{R}(\mathrm{S}(x),\mathrm{S}(v_i))=1\right] \\ &\approx 1-\prod_{i=1}^{N}\Pr[\mathrm{R}(\mathrm{S}(x),\mathrm{S}(v_i))=0] \\ &\approx 1-(1-FAR)^N \\ &\approx N\cdot FAR \end{aligned} \tag{2}$$

The methodology of the fingerprint brute-force attack is to make unlimited submissions of high-quality and liveness fingerprints until success. We donate the success rate after T attempts as *brute-force acceptance rate* (BAR):

$$
\begin{aligned}
BAR :&= \Pr\left[\bigvee_{k=1}^{T} \mathrm{M}_{r=N}(\mathrm{S}(f_k)) = 1\right] \\
&\approx 1 - \prod_{k=1}^{T} Pr\left[\mathrm{M}_{r=N}(\mathrm{S}(f_k)) = 0\right] \\
&\approx 1 - (1 - FPIR)^T \\
&= 1 - (1 - N \cdot FAR)^T
\end{aligned}
\tag{3}
$$

where $T \le \min\{AL, TL \cdot FIPS\}$, AL, TL, $FIPS$ represent the attempt limit, the time limit and the number of submitted fingerprint images per second. On off-the-shelf smartphones, $5 \le AL \le 50, 0 \le TL \le 72 \times 60 \times 60, 1 \le FIPS \le 5$, typically.

According to Equation 3, we can claim that the fingerprint brute-force attack is feasible if adversaries have the following abilities:

A1. Adversaries can bypass the attempt limit AL so that T and $1/FAR$ are in the nearby order of magnitude.

A2. Adversaries can attack fingerprint sensor S to submit images directly. The compromised fingerprint sensor is denoted as $\tilde{\mathrm{S}} : \mathcal{P} \to \mathcal{P}$, which satisfies $\tilde{\mathrm{S}}(p) = p$ for $\forall p \in \mathcal{P}$.

A3. Adversaries can guarantee that each injected image p is high-quality and liveness, that is $\mathrm{Q}(p) = 1$, $\mathrm{L}(p) = 1$, and has at least the $\widetilde{FAR}$ probability to match one of $\mathcal{V}$:

$$
\widetilde{FAR} := \Pr\left[\mathrm{R}(p, \mathrm{S}(v)) = 1\right] \ge \frac{1 - (1 - BAR)^{\frac{1}{T}}}{N}
\tag{4}
$$

3.2 Attack Strategy and Scenario

The threat model of InfinityGauntlet assumes adversaries can physically access the victim's smartphone for a while. Besides, they need to remove the rear cover of the smartphone to plug the adversarial equipment without powering it off and then do some physical manipulations. After tens of minutes or hours of attacks, they can potentially unlock the phone, login privacy apps, complete the payment authentication, and as a by-product, obtain a fingerprint image with similar features to the one enrolled by the victim.

The attack scenario would be expected when the smartphone is lost, stolen, temporarily deposited, or unattended during sleeping.

4 Overview Of InfinityGauntlet

As shown in Fig. 2, the attack surface of InfinityGauntlet is the SPI connected to the fingerprint sensor, where we deploy an SPI MITM framework that can intercept and inject fingerprint images. The attack is divided into two stages: (1) dictionary generation; (2) continuous attempts. The goal of the first stage is to generate a brute-force fingerprint dictionary. This phase intercepts base images from the SPI; then composites these images to a prepared fingerprint database to generate a dictionary. The second stage is responsible for continuously injecting image samples from the dictionary in units of attempt into the SPI. In order to bypass the attempt limit, InfinityGauntlet exploits one of CAMF and MAL vulnerabilities depending on the phone model. CAMF exploitation injects error (e.g., invalidates the checksum) in the last sample to cancel failed attempt, and MAL exploitation infers matching results through side-channel leakage (e.g., duration of unlocking animation) in lockout mode.

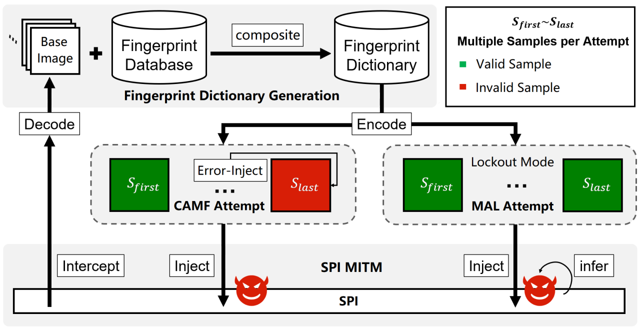


Figure 2: Overview of InfinityGauntlet.

5 Methodology

Three challenges should be tackled to realize our approach: **attempt limit bypassing**, **SPI MITM**, and **fingerprint dictionary generation** which correspond to the abilities of adversaries: **A1**, **A2** and **A3** respectively.This section introduces generic ways to implement them.

5.1 Attempt Limit Bypassing

This subsection details CAMF and MAL that can be exploited to bypass the attempt limit.

5.1.1 Cancel-After-Match-Fail Vulnerability

In biometric authentication systems, capturing multiple image samples in a single authentication attempt is considered one of the best practices for tolerating fault. [31]. Two rules in it are related to CAMF vulnerability:

(1) *Multi-sampling.* To tolerate the false rejection rate of matching algorithms, an attempt can pass the authentication if one of its samples passes.

(2) *Error-cancel.* To tolerate some recoverable errors (e.g., caused by a temporary hardware malfunction), a failed attempt should be canceled when these errors occur.

The multi-sampling mechanism and the exploitation of CAMF vulnerability are shown in Fig. 3. Due to multi-sampling, the authentication for one attempt needs to capture multiple fingerprint samples and then match them in a loop until one sample's result is *error/matched/non-live* or the

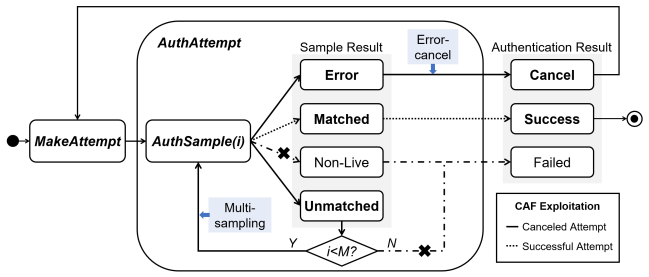


Figure 3: Multi-sampling and CAMF vulnerability.

number of *unmatch* sample reaches the maximum M. Unfortunately, due to error-cancel, adversaries can make unlimited attempts if they fool the M-th sample of each attempt to go into the error-cancel branch by maliciously injecting an error. So the authentication result is either *cancel* or *success*, depending on whether a matched sample is in the first $(M-1)$ samples. In contrast, the *failed* result is always avoided by canceling.

Different smartphone models have different types of errors for canceling, which we will discuss in Section 8.3.

5.1.2 Match-After-Lock Vulnerability

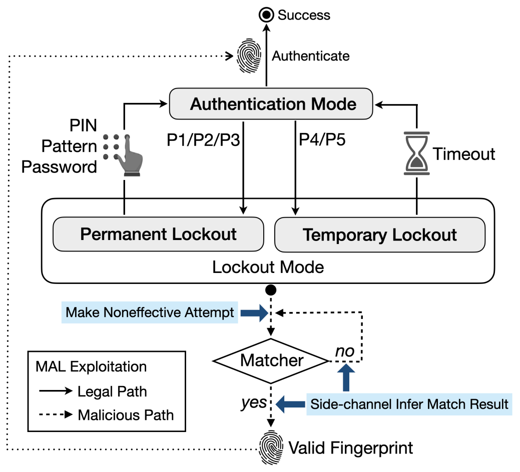


Figure 4: Lockout mode and MAL vulnerability.

In SFA systems, two lockout modes (permanent and temporary) are designed to protect the system when triggering security policies (see Section 2.2). As shown in Fig. 4, the permanent lockout mode is triggered by **P1**, **P2** or **P3**, where fingerprint authentication functions are locked and can only be resumed by passing a primary authentication (e.g., PIN, pattern, password). And the temporary lockout mode is triggered by **P4** or **P5**, where fingerprint authentication functions are locked temporarily and will be resumed when timeout.

However, some manufacturers allow users to make noneffective attempts in lockout modes to improve the user experience when waking up the locked screen by pressing the fingerprint sensor area. The noneffective attempts include all SFA processes except the final authentication action and the increment of failed attempt counter. This user-friendly design may lead to leakage of matching results. As we have mentioned in Fig. 3, if samples in one attempt are all liveness and quality, then the early exit of the attempt means it is a successful authentication attempt. Therefore adversaries can infer the result from side-channel information on response time. For example, a shorter unlock animation or a shorter SPI signal means at least one valid fingerprint exists in the attempt.

As shown in Fig. 4, adversaries first intentionally let the SFA system enter a lockout mode by triggering a security policy (**P1**-**P5**); then make unlimited noneffective attempts in lockout mode until a matched result is inferred by side-channel; lastly, replay the successful attempt that contains the valid fingerprint after the lockout mode is exited to authentication mode by waiting for the timeout or passing a primary authentication.

Since the time window of the permanent lockout mode is infinitely long, the number of MAL attempts is unlimited, so the correct fingerprint can be obtained by brute force. A practical attack can only be performed after exiting the lockout mode, so these attacks are marked COND. However, P40 and Mate30 can repeatedly enter temporary lockout mode to make MAL attempts. They can also obtain an infinite time window and inject the correct fingerprint when the lockout mode automatically exits due to timeout, so these attacks are marked FULL.

In practice, we find two ways to exploit MAL. The first is making noneffective attempts in permanent lockout mode. Although this method can get a correct fingerprint, it cannot unlock the phone immediately. The second way is to make the phone repeatedly enter temporary lockout mode to make noneffective attempts. This way can not only obtain a correct fingerprint but also can inject it when the lockout mode automatically exits due to timeout.

5.2 SPI MITM

This subsection details the SPI MITM, which can implement fingerprint image hijacking.

MITM Attacks on SPI. In SFA systems, serial peripheral interface (SPI) is used for raw fingerprint image transfer between the fingerprint sensor and the smartphone processor. Specifically, the SPI is a four-wire bus consisting of a master input or slave output (MISO), a master output or slave input (MOSI), a serial clock (SCLK), and a chip select (CS), where the smartphone processor acts as the *master* and the fingerprint sensor acts as the *slave*. Typically, the SCLK has a frequency range of 8~50 MHz, and a checksum is used for error-checking. However, to our knowledge, almost all SFA sensors, except Touch ID, do not encrypt image data and lack mutual authentication, making SFA systems vulnerable to MITM attacks on SPI.

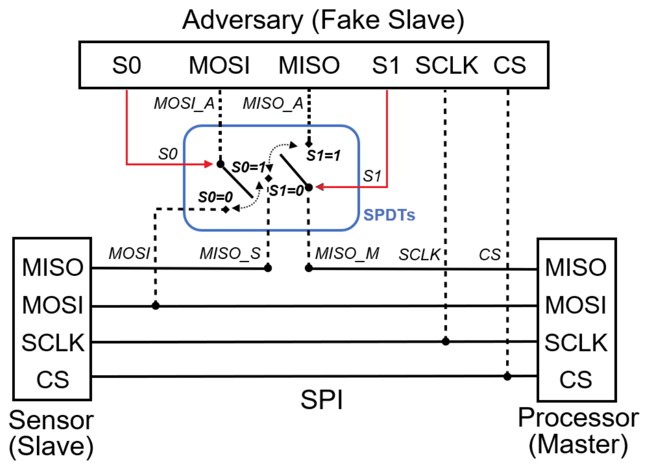


Figure 5: MITM Attack on SPI.

We design the SPI MITM of SFA in Fig. 5. The adversary acts as a *fake slave* to receive the MOSI_A signal or send the MISO_A signal according to its working mode that is controlled by two single pole double throw (SPDT) switches: *S0* and *S1*. Different states of SPDTs are explained in Table 1. *Idle mode* is activated when $S0 = 0$ and $S1 = 0$, in which the adversary listens for the fingerprint data acquisition (FDA) command. *Intercept mode* is activated when $S0 = 1$ and $S1 = 0$, in which the adversary intercepts the raw fingerprint image. *Inject mode* is activated when $S0 = 0$ and $S1 = 1$, in which the adversary injects the raw fingerprint image. The MITM framework implements raw fingerprint image intercept and injection through mode switching.

A highlight of this framework is to locate the fingerprint image transmission window through FDA and only hijack the data in the window so as not to affect other interactive processes.

Table 1: Function of the two SPDT switches.

State		Function
S0	0	Identify the FDA command from *MOSI*.
	1	Intercept raw image from *MISO_S*.
S1	0	Keep connection from *MISO_S* to *MISO_M*.
	1	Inject raw image from *MISO_A* to *MISO_M*.

Raw Image Decoding and Encoding. The MITM framework also requires functions for decoding and encoding the raw fingerprint image. The two functions are inverses and fit the transmission protocol to deal with image shape, pixel representation, byte sequence, checksum, and frame structure. For instance, necessary steps for the encoding function include pixel adaptation to unify bits per pixel, byte reordering to compress adjacent pixels, checksum attachment to verify image data, and frame alignment to separate and add a header to frames. We will show a running example of encoding/decoding in Section 7.2.

5.3 Fingerprint Dictionary Generation

In this subsection, we detail how to generate a fingerprint dictionary containing a large number of fingerprints for brute-force attacks. We solve this problem by proposing a synthetic fingerprint generation method for SFA for the first time. The method includes device-specific base image capturing and device-independent fingerprint composition.

5.3.1 Base Image Capture

As authentication workflow described in Fig. 1, SFA performs compensation on the raw fingerprint image and then does the quality check and liveness detection. If the raw fingerprint image submitted by adversaries does not have a background that matches the device, it will be rejected by quality check or liveness detection. Therefore, we need to collect the device-specific base image from the victim's phone.

To avoid capturing the texture of the pressed object, we use a so-called hollowed-out press to trigger finger sensing. The hollowed-out press refers to using a hollow and conductive fake finger to press the fingerprint sensing area. Since the finger sensing of SFA only relies on the total area pressed, the hollowed-out press can also trigger image capturing if the hollowed-out area is not too large. Besides, we use the attempt limit bypassing method mentioned above to perform multiple hollowed-out presses to obtain multiple base images through SPI MITM. We will show the hollowed-out press for an in-display optical fingerprint smartphone in Section 7.3.

5.3.2 Fingerprint Composition

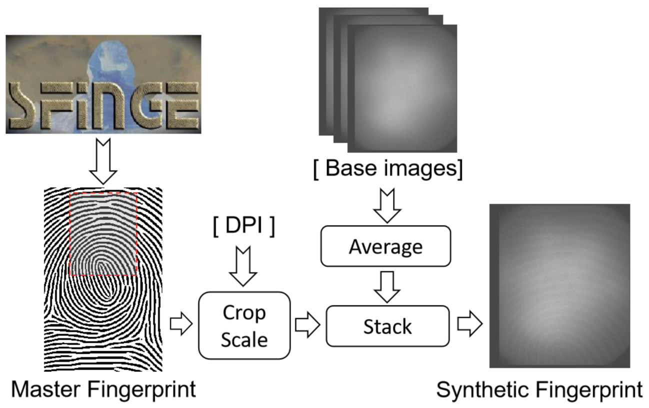


Figure 6: Synthetic fingerprint generation for SFA.

We propose a fingerprint composition method for SFA based on the *master fingerprint*, an intermediate product of

SFinGe [22]. SFinGe is a state-of-the-art synthetic fingerprint generation method for evaluating the *FAR* of matching algorithms. The *master fingerprint* is a binary fingerprint image containing only valley-ridge patterns but includes the core information to distinguish different fingerprints.

To make the synthetic fingerprints suitable for brute-force attack, we make the following adaptation to SFinGe, as shown in Fig. 6. Firstly, we use the pure *master fingerprint* that does not impose intra-fingerprint variations (such as ridge thickness variations, distortions, perturbations, and global translation/rotation) on it. That is because the variations are counterproductive to increasing the success rate of brute force attacks. Secondly, we crop and scale the *master fingerprint* depending on the DPI (Dots per Inch) measured from the victim's phone. Finally, we average multiple base images pixel by pixel to reduce noise.

We provide a running example of how to generate in-display optical fingerprints in Section 7.3.

6 Adversarial Equipment

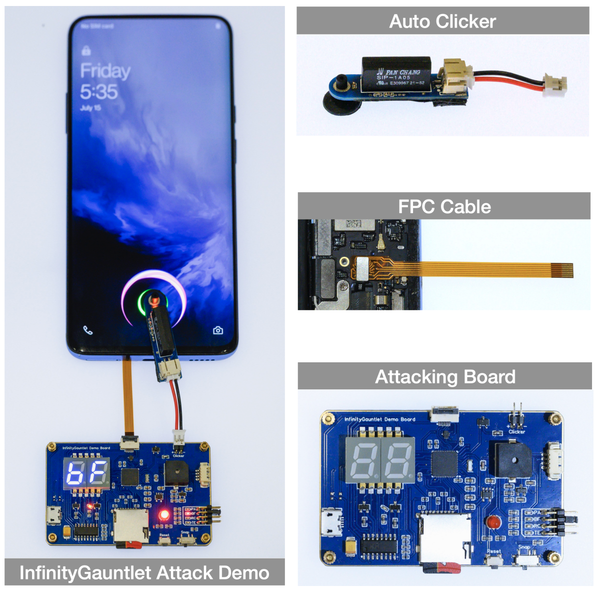


Figure 7: Adversarial equipment and attack demonstration.

The adversarial equipment we implemented for INFINITYGAUNTLET includes *attacking board*, *auto clicker*, and *flexible printed circuit (FPC) cable*. The *attacking board* and *auto clicker* are general for different smartphones, only the *FPC cable* needs to be customized based on the target smartphone model. Fig. 7 shows the attacking demonstration and the equipment components.

Attacking Board. *Attacking board* is the core of the equipment which consists of four main components: (1) STM32F412: an MCU that has an SPI peripheral with 38 MHz transmission rate in slave mode that controls the whole attack process; (2) RS2117: an SPDT analog switch with 400 MHz bandwidth that switches between working modes of SPI MITM; (3) SD Card: an 8GB flash memory that can store a fingerprint dictionary containing up to 200 000 fingerprint images; (4) Pin Header Switch: a pin header array that can switch between different functions, such as raw image capture (RC), brute-force attack (BF), template enrolls (TE), and presentation attack (PA) by a jumper cap.

Auto Clicker. *Auto clicker* is designed to simulate finger pressing physically. The core of it is a reed relay SIP-1A05 connected to a conductive suction cup attached to the sensing area. *Auto clicker* simulates pressing/lifting by opening/closing the relay, so the frequency of clicks can be controlled by *attacking board*.

FPC Cable. *FPC Cable* is responsible for taking signal lines such as CLK, CS, MISO_M, MOSI, and MISO_S from the smartphone's motherboard to *attacking board*. *FPC Cable* is connected to phone by a B2B socket, so no welding is required when attacking. Due to the different B2B sockets of different fingerprint sensors, the circuit of *FPC cable* needs to be customized based on the target phone model.

The total equipment costs less than 20 dollars. **In order to fill the gap in the research tools of SFA, we decided to open source all PCB (printed circuit board), BOM (bill of materials), and firmware to the academic community**[1].

7 A Case Study of INFINITYGAUNTLET

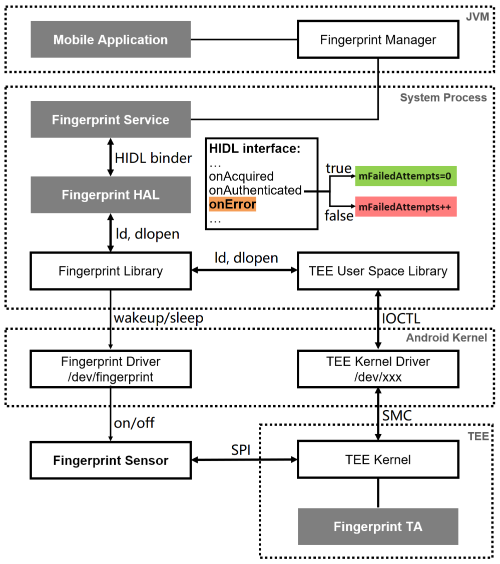
In this section, we detail a concrete case study on OnePlus 7 Pro (called the "victim device" throughout the section) and present the result of brute-force attacks. We explain why choosing the victim device as a case study. (1) It runs close to the stock version of Android, which is open-source and the most popular, making it representative for studying SFA. (2) Its fingerprint sensor uses the in-display optical technology that has become the standard configuration of most mid-end and high-end smartphones.

7.1 Attempt Limit Bypassing

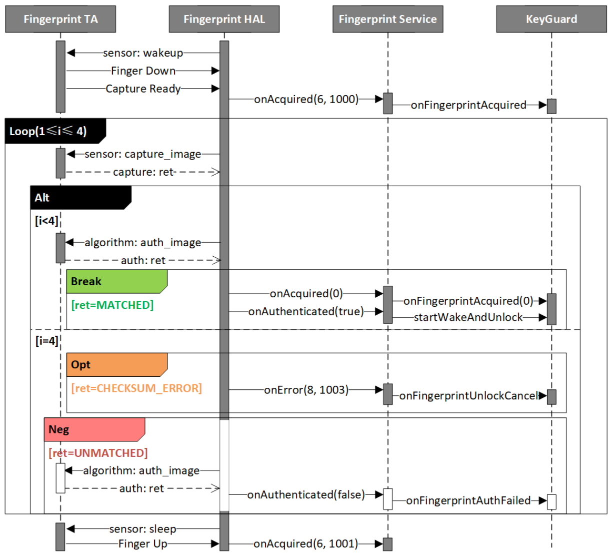
We find CAMF vulnerability on the victim device and make exploitation with the checksum *error-cancel*. Unlimited attempts can be achieved with CAMF exploitation.

Android SFA Framework. The victim device uses a fingerprint authentication framework of Android, as shown in Fig. 8(a). Each *mobile application* has its own *fingerprint manager* to communicate with *fingerprint service* which is implemented by a class *FingerprintService* that belongs to android open source project (AOSP). *Fingerprint service* connects to *fingerprint HAL* with *HIDL interface* to communicate with vendor-specific *fingerprint libraries*. In *TEE*, *fingerprint TA* runs core algorithms including compensation, anti-faking

[1] https://github.com/alohachen/InfinityGauntlet



(a) SFA framework in Android.



(b) Sequence diagram for attempt limit bypassing by CAMF.

Figure 8: Explanation of the attempt limit bypassing in the case study. In Android biometric framework, fingerprint authentication is closely related to four roles: *Fingerprint TA*, *Fingerprint HAL*, *Fingerprint Service*, and *Mobile Application* (i.e, KeyGuard). The interactions between them under our attack are represented with a sequence diagram.

and matching; and *TEE kernel* communicates with *fingerprint sensor* through *SPI*.

Three important *callback*s are defined by the framework in *HIDL interface* to return asynchronous results from *TEE*: (1) *onAcquired* is called back when a fingerprint image is captured by the sensor and notifies a *acquiredInfo* message about the image quality. (2) *onAuthenticated* is called back when a fingerprint is authenticated and notifies an authentication result. (3) *onError* is called back when a fingerprint error occurs and notifies a *error* message.

The callbacks are implemented in *FingerprintService*, where the attempt limit is monitored by a variable named *mFailedAttempt* and can only be increased when *onAuthenticated* is called back, which will happen when *Fingerprint TA* request *Fingerprint HAL* to notify authentication results. If the authentication result is *FALSE*, *mFailedAttempt* will be increased by one, which decreases the remaining number of available attempts. However, *mFailedAttempt* can remain unchanged if adversaries construct an error event to trigger *onError* for bypassing *onAuthenticated*.

Checksum Error-cancel. As checksum is recommended for SPI transmission protocols to do error-check, most SFA system attaches it to fingerprint data and regard a recoverable error as a cancelable state for authentication. To exploit CAMF on the victim device, we invalidate the checksum of the fourth sample's fingerprint data and successfully trigger the error-cancel rule.

We extract the control flow of the victim device's fingerprint authentication framework to reason the CAMF exploitation and show it in Fig. 8(b). Once *fingerprint HAL* receives an authentication request, it will send a wakeup command to *fingerprint TA* and notify the *onAcquired* callback to *fingerprint service*. Then the authentication loop goes through image capturing and authentication for each sample until the last one which notifies the *onError* callback with a vendor-specific code (1003) to *fingerprint service*. It is worth mentioning that under our attack, the loop never ends with an unmatched sample result to notify *onAuthenticated* with a failed authentication result. Moreover, if one of the first three sample is matched, *fingerprint HAL* will notify *onAcquired* with the code 0 (*FINGERPRINT_ACQUIRED_GOOD*), send a success result to *onAuthenticated*, and directly break the loop.

7.2 SPI Protocol Reverse Engineering.

We observe SPI signals through a logic analyzer and locate those dense signals on MISO to identify fingerprint image data. As the data is not encrypted, we can try out the encoding method with some tricks. For example, the image shape can be guessed by factoring the total number of pixels, and adaption can be made according to the periodic offset of outlier values (i.e., values such as checksum other than image pixels). We find the first sample is transmitted in 4 frames, while the last three use the same format with 13 frames. Each last frame is short since it transmits the remained fingerprint data. The FDA commands can be identified before every frame. Take the first sample as an example: the FDA commands are al-

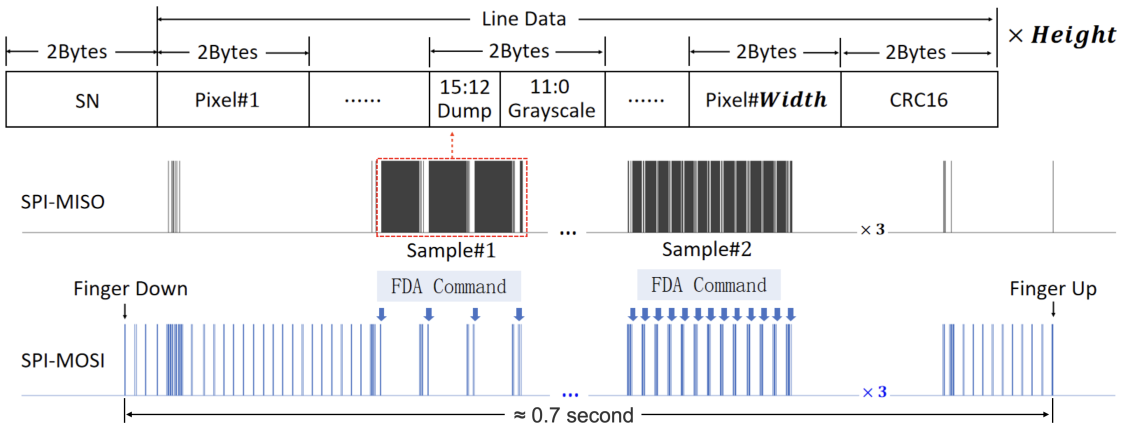


Figure 9: Example of reverse-engineered SPI protocol for fingerprint data on MISO and MOSI.

ways $0xF08800$, and we show the structure (frame separator omitted) of fingerprint data in Fig. 9. The structure is not complicated: the gray-scale image is stored in 16 bits per pixel, and for each line, a serial number and a CRC16 checksum are attached at both ends. It is worth noting that our method only operates on the data within the frame, so there is no need to reverse the full interactive protocol.

7.3 Fingerprint Dictionary Generation

Based on the CAMF and SPI MITM mentioned above, adversaries can perform INFINITYGAUNTLET if they have a valid fingerprint dictionary. This subsection shows how to generate a fingerprint dictionary for the victim's device.

Base Image Capture. We take a rubber, hollowed out a small piece on the bottom side, as shown in Fig. 10, and paint the surface with black conductive paint to make a hollowed-out fake finger. Then press it on the fingerprint sensing area and apply SPI MITM and CAMF to capture base images. Since the victim device applies two different exposure times for different samples in one attempt, so the base image is classified into two brightness levels, as shown in the lower left part of Fig. 11. To reduce noise, we collect multiple base images for Sample#1 and Sample#2/#3/#4, respectively, then average them by pixel to get the final base images. The base images include regular grids and noticeable vignetting effects. We guess these fixed patterns are introduced by the screen pixel diffraction and the lens limitations of the in-display optical fingerprint sensor.

DPI Measurement. We take a piece of conductive rubber and carve a short line of known length L on it. Then press it on the fingerprint sensing area to capture the image, and count the number of pixels P occupied by the short line. Finally, divide P by L to get DPI. For example, in the experiment we did on the victim device: 0.237 inch corresponds to 154 pixels, so the calculated DPI is 652 (154/0.237).

Master Fingerprint Generation. We use Anguli [3], an

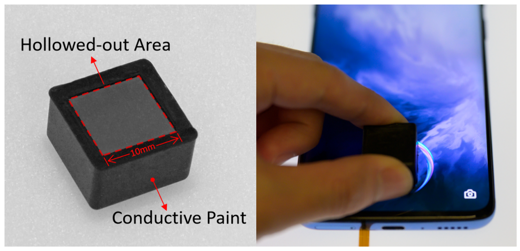


Figure 10: Press a hollowed-out and conductive rubber on the fingerprint sensing area to capture base images.

open source implementation of SFiGe, to generate 200,000 *master fingerprints* in default parameters. The *master fingerprint* generated by Anguli has a resolution of 275x400 with 500 DPI.

Crop and Scale. Since the two DPIs are not match, *master fingerprint* needs to be scaled according to their ratio (652/500). Next, we need to crop the scaled *master fingerprint* to the same size (192x224) as the base image. Considering the common user pressing habits, we prefer to crop the upper middle part of *master fingerprint*, as shown in the left part of Fig. 6.

Fingerprint Composition. The final step is to composite fingerprints. As shown in Fig. 11, we first apply the ROI mask to *master fingerprints* to simulate the vignetting effects of the lens. Then stack them with the base image to simulate the pixel diffraction effect of the in-display fingerprint.

7.4 Experiment Results

We design brute-force experiments on screen unlock to evaluate INFINITYGAUNTLET for this case study. As mentioned in Section 3, the effect of InfinityGauntlet is related to the number of fingers enrolled by the victim. So we experiment on two extreme scenarios: enroll one finger and five fingers. We generate a fingerprint dictionary containing 200000 im-

Table 2: Results of brute-force experiments on two enrollment scenarios. The "Success #" represents the average number of successes when 200,000 synthetic fingerprints are injected twelve times. The "Failed #" represents the average number of failures that caused *mFailedAttempt* to increase. The "Time Cost" is the average time required for one successful attack. The "Attack *FAR*" is the practical *FAR* of injected fingerprint images. The "Baseline *FAR*" is the referenced *FAR* calculated by Equation 4, that is required for a successful brute force with an 0.8 probability (*BAR*=0.8) within 36 hours (TL=36 $\times 60 \times 60$).

	Enroll One Finger	Enroll Five Fingers
Success #	9.2[1]	41.9[2]
Failed #	0	0
Time Cost	2.01 hours	0.44 hours
Attack *FAR*	4.6×10^{-5}	4.2×10^{-5}
Baseline *FAR*	4.14×10^{-6}	0.83×10^{-6}

[1] 9.2 = mean(8, 11, 8, 7, 15, 7, 8, 13, 8, 6, 12, 7)
[2] 41.9 = mean(38, 42, 43, 48, 42, 44, 40, 43, 34, 48, 40, 41)

ages. Using the dictionary, we conduct twelve brute-force experiments for the two scenarios at $FIPS = 3$ (3 fingerprint images are injected per second). Each enrolled fingerprint differs from experiment to experiment and does not appear in the dictionary.

Table 2 shows the statistical metrics and average results. The "Failed #" metric is always zero, meaning the attempt limit (*AL*), quality check (Q) and liveness detection (L) that defined in Section 3 are all compromised completely. The victim device is successfully unlocked 9.2 times when "Enroll One Finger" and 41.9 times when "Enroll Five Fingers"; the corresponding "Time Costs" are 2.01 and 0.44 hours, respectively. We notice something interesting: the "Attack *FAR*" is at least an order of magnitude higher than the "Baseline *FAR*", suggesting that INFINITYGAUNTLET has strong robustness in

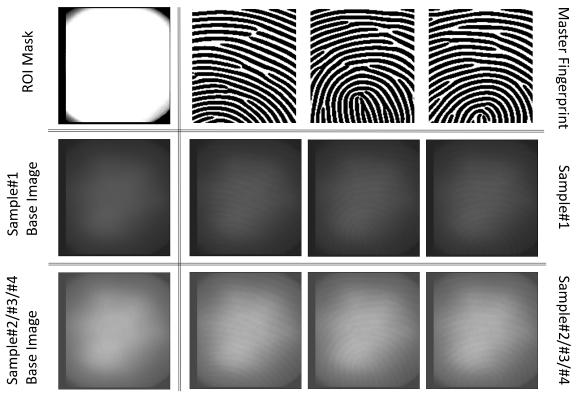


Figure 11: Examples of fingerprint composition.

this case study. The TL we choose for the baseline is 36 hours because this is the mathematical expectation of TL when the adversary controls the phone according to **P2**. Fig. 12 shows four samples randomly selected from the injection fingerprints of successful attacks.

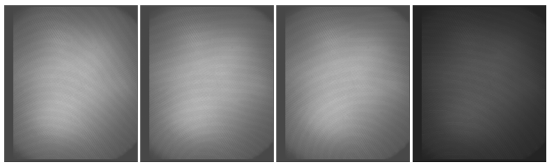


Figure 12: Successful samples from the experiment.

8 Scalability of INFINITYGAUNTLET

After confirming the feasibility of INFINITYGAUNTLET through the case study, we begin to discuss the scalability of our attack in this section.

8.1 Target Selection

Scalability experiments are done on 12 off-the-shelf smartphones. We cover mainstream smartphone manufacturers and sensor vendors on the market with our best effort, although similar issues found on newer Android 12 devices cannot be disclosed due to NDA. Besides the OnePlus discussed in the case study, we select smartphones from top manufacturers [11]: Samsung, Huawei, Xiaomi, Oppo, Vivo, OnePlus, Honor, and Apple. The specific models are chosen according to their fingerprint sensors and OSs. For the scanning technology of SFA sensors, the pioneer is capacitive scanning. The representative is Touch ID, so we pick two Apple models. We also pick an Android model equipped with a capacitive sensor to complement. The increasing applications of in-display optical sensors are the latest trend, where Goodix dominates the market and adopts optical scanning. Hence, we choose the maximum number of models equipped with them. We select Mi 11 Ultra since it represents the most current ultra-thin in-display optical sensor. A branch of in-display sensors adopts ultrasonic scanning developed by Qualcomm, for which we choose two Samsung Galaxy series. Two models from Huawei are also selected since they use the recently released HarmonyOS and embed fingerprint sensors supplied by different vendors. We evaluated typical fingerprint authentication applications, including screen lock, payment, and privacy.

However, due to time constraints, it is difficult for us to conduct a complete experiment like the case study in Section 7 for all test targets. Therefore, we choose to verify whether the test phones have CAMF/MAL and SPI-MITM vulnerabilities and then infer the possible attack types according to the

Table 3: Experimental smartphones, sensor information, discovered vulnerabilities, and the attack types on three typical fingerprint authentication applications. ✔ means the vulnerability exists. ✘ means the vulnerability does not exist. ◯ means the vulnerability exists but has a finite effect.

Smartphone		Sensor		Vulnerability			Attack		
Manuf./Model	OS/Ver.	Vendor	Type	CAMF	MAL	MITM	Unlock[1]	Payment[2]	Privacy[3]
Samsung Galaxy S20U	Android 11	Qualcomm	Ultrasonic	✔	✘	✔	*FULL*	*FULL*	*FULL*
Samsung Galaxy S10+	Android 9	Qualcomm	Ultrasonic	✔	✘	✔	*FULL*	*FULL*	*FULL*
Xiaomi Mi 11 Ultra	Android 11	Goodix	Optical	✔	✘	✔	*FULL*	*FULL*	*FULL*
OnePlus 7 Pro	Android 11	Goodix	Optical	✔	✘	✔	*FULL*	*FULL*	*FULL*
OnePlus 5T	Android 8	Goodix	Capacitive	✔	✘	✔	*FULL*	*FULL*	*FULL*
Huawei Mate30 Pro	HarmonyOS 2	Goodix	Optical	✘	✔	✔	*FULL*	*FULL*	*FULL*
Huawei P40	HarmonyOS 2	Novatek	Optical	✘	✔	✔	*FULL*	*FULL*	*FULL*
OPPO Reno Ace	Android 10	Goodix	Optical	✔	✘	✔	*FULL*	*FULL*	*FULL*
Honor Magic3	Android 11	Goodix	Optical	✘	✔	✔	*LIMIT/COND*	*FULL*	*FULL*
Vivo X60 Pro	Android 11	Goodix	Optical	◯	✔	✔	*LIMIT/COND*	*FULL*	*LIMIT*
Apple iPhone 7	iOS 14.4.1	AuthenTec	Capacitive	◯	✘	✘	*PA_ONLY*	*PA_ONLY*	*PA_ONLY*
Apple iPhone SE	iOS 14.5.1	AuthenTec	Capacitive	◯	✘	✘	*PA_ONLY*	*PA_ONLY*	*PA_ONLY*
Apple iPhone SE(2nd)	iOS 15.5	AuthenTec	Capacitive	◯	✘	✘	*PA_ONLY*	*PA_ONLY*	*PA_ONLY*

[1] Unlock: screen unlock.
[2] Payment: make payments on pre-installed or third-party payment apps, including: Paypal, Alipay, Samsung Pay, Huawei Pay, OPPO Pay, Vivo Pay, and Apple Pay.
[3] Privacy: log into pre-installed privacy protection apps, including Secure Folder for Samsung, Hidden Folders for Xiaomi, LockBox for OnePlus, Safe for Huawei and Honour, Private Safe for OPPO, File Safe for Vivo and Notes for Apple.

degree of vulnerability and the exploitability in different applications. The logic behind this choice is that the existence of CAMF/MAL and SPI-MITM means that the abilities **A1** and **A2** discussed in the threat model are satisfied. Meanwhile, because the SFA algorithms developed by vendors have strong homology, we assume that the ability **A3** can also be satisfied.

8.2 Empirical Analysis Result

The empirical analysis results are given in Table 3. We identify four types of attacks with different strengths: *FULL*, *LIMIT*, *COND* and *PA_LIMIT*. The *FULL* attack means the adversary can perform a full INFINITYGAUNTLET attack. The *LIMIT* attack means the adversary can perform a limited attack, that the attempt limit can be bypassed but can not enlarge to infinite. The *COND* attack means the adversary can perform a conditional attack, where the valid fingerprint image obtained by brute force needs to be re-injected after the lockout mode is exited by primary authentication. The *PA_ONLY* attack represents that the adversary can only perform a presentation attack with the expanded number of attempts because fingerprint injection via SPI-MITM is impossible, and the attempt limit can not enlarge to infinite. From Table 3, we can see all models but iPhones have at least two vulnerabilities, and at least one fingerprint authentication application is subject to the full INFINITYGAUNTLET attack. **We have submitted these vulnerabilities to these seven manufacturers, and all have been confirmed, including critical and high ones. After we submitted these vulnerabilities, Google also raised the security requirements of the "false trial" in the Android compatibility definition document (CDD) [2] to prevent fingerprint brute-force attacks**.

In fact, after submitting this paper, we still found an exploitable CAMF vulnerability on Motorola S30 Pro which was launched on August 2022 by Lenovo, with Android 12.

In addition, we open the black box of the SFA through SPI MITM and expose fingerprint images of different types of sensors for the first time, as shown in Fig. 13. Mate 30 Pro 5G and OnePlus 7P use the same pixel encoding method, both are 13bit grayscale images with resolutions of 182x182 and 192x224, respectively. Due to the capacitive technology, the image of OnePlus 5T has a smaller resolution, which is a 108x88 12bit grayscale image. The advanced ultra-thin in-display optical fingerprint technology adopted by Mi 11 Ultra realizes color sense by adding a color filter to the sensor and captures a 126x122 color image in 32bit CMYK format. The image of the Galaxy S10+ is composed of several 108x80 12-bit grayscale sub-images vertically. Due to the use of ultrasonic technology with a high-noise floor, it is not easy to see the fingerprint pattern on the raw image. So we use the collected base images to compensate for the raw image (left half), and a clear fingerprint pattern can be seen in the processed image (right half). In summary, we successfully obtained the raw fingerprint images by reverse engineering from the intercepted SPI data for each type of fingerprint sensor. Moreover, as Mate 30 Pro and OnePlus 7 Pro use in-display optical fingerprint sensors provided by Goodix, we find the protocols share much in common. So we guess there will not be much work for the protocol reverse on untested smartphone models since there are only a handful of sensor vendors on the market.

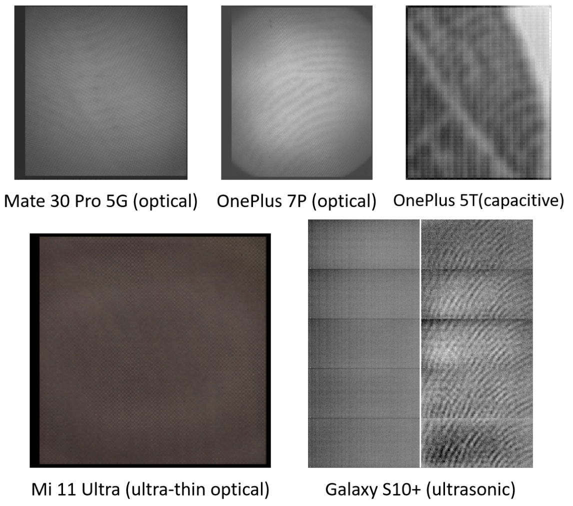


Figure 13: Fingerprint raw images intercepted by SPI-MITM.

8.3 Details and Findings

This subsection supplements some technical details of the empirical analysis and shares some interesting findings.

Sensor Hot Plugging. Because the FPC cable needs to be inserted between the fingerprint sensor's B2B sockets, INFINITYGAUNTLET requires the smartphone to support sensor hot plugging. Otherwise, restarting the smartphone will trigger the security policy **P3**. Therefore, before experimenting with each smartphone, we firstly checked whether its sensor supports hot plugging. Results suggest that all smartphones support hot plugging. We guess this is the default feature across the industry.

Different Error-cancels. In our experiments, we find that the *error-cancels* triggering CAMF involve various errors: CRC check error (E1), too fast lifting error (E2), timeout unresponsive error (E3), lock screen interrupt error (E4). For example, Galaxy and the Mi 11 Ultra use E1 and E2; OnePlus uses E1 and E3; Reno Ace and X60 Pro utilize E4. Exploiting these errors for CAMF follows the methodology we summarized in Section 5.1.

Restricted Attempt Limit Bypassing. From Table 3, for some applications, only restricted attempt limit bypassing can be achieved on X60 Pro and iPhones, and their attempt limit can be enlarged from 20 to 50 and from 5 to 15, respectively. We guess that the X60 Pro and iPhones introduced additional counters to limit the number of failures.

MAL Exploitation in Different Lockout Modes. In the experiments, we find that Mate30 Pro, P40, Magic3, and X60 Pro are affected by MAL, but they can be exploited in different lockout modes by triggering different security policies. Mate30 Pro and P40 can be exploited in temporary lockout mode triggered by **P5**. Magic3 can be exploited in temporary lockout mode triggered by **P4**. Magic3 and X60 Pro can be exploited in permanent lockout mode triggered by **P1/P2/P3**. Due to the different lengths of time windows available for making noneffective attempts (see Fig. 4) and different lockout mode exiting conditions, these attacks have different strengths.

9 Mitigation

To defend against the attempt limit bypassing based on CAMF, we recommend checking each attempt to see whether a cancellation happens and switching to "No Cancel Mode" once the historical cancel numbers reach a threshold. In "No Cancel Mode", the smartphone can only be unlocked when there is no canceled sample in the current attempt. The mitigation balances usability with security as the unlocking time is only increased slightly in "No Cancel Mode" while the false reject rate is not increasing. To prevent SPI MITM, we suggest that fingerprint sensor vendors and smartphone manufacturers are responsible for encrypting crucial data during communication. Most importantly, the pin MISO should carry high entropy data. Besides, we suggest that the fingerprint acquisition behave consistently in attempts so that adversaries cannot infer the matching results by side-channel information. For example, at the UI level, preventing the animation of fingerprint acquisition leaks the authentication results. Lastly, we recommend moving the attempt limiting logic from REE to TEE to protect against privileged adversaries. Therefore thoroughly addressing InfinityGauntlet requires the whole industry to be aware of our proposed attack model.

10 Discussion

We discuss the potentials of INFINITYGAUNTLET's technique beyond fingerprint brute-force attack, the limitations of INFINITYGAUNTLET, and our future work.

Attack Beyond Brute-force. Besides achieving fingerprint brute-force attacks that require no prior knowledge about the victim, the proposed method can also enhance traditional presentation attacks that require the fingerprint latent of the victim. The first point is that the attempt limit is no longer a barrier for smartphone presentation attacks. More importantly, the laborsome fabrication process can be replaced with image-level editions, which can significantly improve *SAR*.

Biometric Beyond Fingerprint. As the multi-sampling mechanism is considered one of the best practices in biometric authentication systems, there is reason to suspect that CAMF also exists in other biometric systems based on face, iris, or palmprint. In fact, from the AOSP code, we find the integration of the biometric authentication framework in *hardware.biometrics.BiometricManager* and *server.biometrics.BiometricServiceBase* packages, where the *error-cancel* mechanism can also be seen. However, since the transmission channel differs among biometric sensors, some hardware-related adaption must be made to confirm the vulnerability to brute-force attacks and make exploitation. Nevertheless, INFINITYGAUNTLET offers a new perspective on the attack surface of various biometric authentication systems.

Limitations. (1) Our work can not bypass the limit of the security policy **P2**, so the full INFINITYGAUNTLET attack must be completed in 72 hours or less. Nevertheless, according to the experimental results of the case study, adversaries can complete the attack in a few hours. (2) The valid fingerprint image guessed by INFINITYGAUNTLET does not look similar to the enrolled finger from a human observation point of view. That is because, as described in Section 2.1, the matching algorithm in SFA is based on template features rather than overall image relevance. (3) Although we have introduced some tricks of inferencing image transfer protocol in Section 7.2, which are applied to most smartphone models, it is not excluded that there are still some difficulties for a few models. For example, we found that some models use a custom checksum algorithm to check each line in the image. Our solution to this problem is to reverse engineer the associated checksum code in the fingerprint TA.

Future Work. The time cost of the attack is related to the distribution of injected fingerprints. However, the fingerprint dictionary used in INFINITYGAUNTLET is non-selective, so there is much room for improvement. The idea from DeepMasterPrint [20] can be borrowed to optimize fingerprints that trigger more collisions. With the help of our open-source adversarial equipment, researchers can easily inject fingerprints into smartphones to evolve the dictionary. If a well-design dictionary is obtained, we believe it is also probable to attack smartphones where the attempt number can not enlarge to infinite.

11 Related Work

Prior works have proposed different PAs targeting traditional fingerprint authentication systems. The most widely adopted methodology is to impersonate a legitimate user via artefacts [29], such as latent print images [21] and gummy fingers [18, 32]. On off-the-shelf SFA systems, a few PAs were reported successful. For instance, the CCC team [26] fabricated a latex sheet after photographing a victim's fingerprint and bypassed the iPhone 5S Touch ID. A recent study from Cisco [36] selected fabric glue to make 3D printed fingers, which claim to bypass 8 out of 13 tested smartphones. These works differ mainly in the chosen materials and fingerprint engraving techniques. In comparison, our work can inject arbitrary fingerprint images by SPI MITM. This way, we avoid delicate crafts and enhance the attack through effortless image editions. Besides, the above attacks all rely on the knowledge of the victim's fingerprint. Whereas our work can make exploitation to achieve infinite fingerprint brute-force attempts, becoming free of prior knowledge about the victim's fingerprint.

Zhang et al. [40] study four types of security pitfalls on SFA that may be exploited by malware and are the first (in 2015) to discuss the attacks targeting the authentication framework. However, as the architectural security of SFA has much enhanced in recent years, especially with securer TEE implementations, follow-on works hardly appear. Our work discovers defects in state-of-the-art SFA frameworks that affect the most advanced software and hardware solutions.

MasterPrint [17] investigates the security of partial fingerprint authentication systems, especially when multiple user fingerprints are enrolled. DeepMasterPrints [20] further demonstrate the vulnerability to collision attacks. However, the proof-of-concept of DeepMasterPrints cannot work in a high security-level setting. Specifically, only 1.11% attacks are shown successfully on the VeriFinger [16] (a none-SFA matcher without liveness detection) with 0.01% *FAR*. In contrast, we manage to attack off-the-shelf smartphone authentication systems in a much tougher 0.002% [1, 9] *FAR* setting with a very high probability of success.

Prior works make efforts to defeat attacks through liveness detection with extra hardware or image analysis [25, 27]. The hardware-based methods use specialized sensors to capture biological characteristics [19, 33]. However, they require additional hardware costs. The image-based approaches leverage features such as texture to separate live and dummy fingerprint images [24], and many recent works are learning-based [34]. Inspired by [23], our work injects fingerprint images directly rather than using an intermediate, which fools image-based liveness detection by nature.

12 Conclusion

This paper proposes a novel fingerprint brute-force attack on off-the-shelf smartphones for the first time. Adversaries can pass fingerprint authentication with zero knowledge of the victim to unlock the smartphone, log into privacy apps and make payments. We hope our work can inspire the industry to improve the security of biometric authentication.

Acknowledgments

The authors thank the anonymous reviewers and shepherd for their careful feedback. We would like to thank Yiling He, Huiming Liu, Zhuo Ma, Ye Li, Ge Dai and Jiangchao Zhang for their help and support. We also thank Huawei, Xiaomi, Honor, Vivo, OPPO, and Samsung for promptly reviewing and fixing the vulnerabilities we submitted. A'Tuin, a security topography system developed by Xuanwu Lab, provides convenience for our vulnerability mining.

References

[1] About touch id advanced security technology. https://support.apple.com/en-us/HT204587.

[2] Android-12-cdd biometric sensors. https://source.android.com/compatibility/12/android-12-cdd#7310_biometric_sensors.

[3] Anguli: Synthetic fingerprint generator. https://dsl.cds.iisc.ac.in/projects/Anguli/.

[4] Better biometrics in android p. https://android-developers.googleblog.com/2018/06/better-biometrics-in-android-p.html.

[5] CVE-2019-17668. https://nvd.nist.gov/vuln/detail/CVE-2019-17668.

[6] CVE-2020-11600. https://nvd.nist.gov/vuln/detail/CVE-2020-11600.

[7] CVE-2020-7958. https://nvd.nist.gov/vuln/detail/CVE-2020-7958.

[8] CVE-2021-22494. https://nvd.nist.gov/vuln/detail/CVE-2021-22494.

[9] Fido biometric performance levels. https://fidoalliance.org/specs/biometric/requirements/#FARReq.

[10] Friend was able to unlock my phone with his fingerprint. https://www.reddit.com/r/galaxys10/comments/b6mnzq/friend_was_able_to_unlock_my_phone_with_his/.

[11] Global smartphone market in the second quarter of 2021. https://www.canalys.com/newsroom/global-smartphone-market-q2-2021.

[12] Huawei security advisory - fingerprint unlocking vulnerability on smartphones. https://www.huawei.com/br/psirt/security-advisories/2018/huawei-sa-20180203-01-fingerprint-en.

[13] Measuring biometric unlock security. https://source.android.com/security/biometric/measure.

[14] My daughter's thumb unlocked my phone. https://www.reddit.com/r/apple/comments/59okc3/my_daughters_thumb_unlocked_my_phone_she_does_not/.

[15] My friend was able to unlock my 6t by using his own fingerprint. https://www.reddit.com/r/oneplus/comments/9vbh8u/my_friend_was_able_to_unlock_my_6t_by_using_his/.

[16] Verifinger. https://www.neurotechnology.com.

[17] Aditi, Roy, Nasir, Memon, Arun, and Ross. Masterprint: Exploring the vulnerability of partial fingerprint-based authentication systems. *IEEE Transactions on Information Forensics and Security*, 2017.

[18] S. S. Arora, A. K. Jain, and N. G. Paulter. Gold fingers: 3d targets for evaluating capacitive readers. *IEEE Transactions on Information Forensics and Security*, 12(9):2067–2077, 2017.

[19] Denis Baldisserra, Annalisa Franco, Dario Maio, and Davide Maltoni. Fake fingerprint detection by odor analysis. In *International Conference on Biometrics*, pages 265–272. Springer, 2006.

[20] Philip Bontrager, Aditi Roy, Julian Togelius, Nasir Memon, and Arun Ross. Deepmasterprints: Generating masterprints for dictionary attacks via latent variable evolution. In *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pages 1–9. IEEE, 2018.

[21] Kai Cao and Anil K Jain. Hacking mobile phones using 2d printed fingerprints. *Dept. Comput. Sci. Eng., Michigan State Univ., East Lansing, MI, USA, Tech. Rep. MSU-CSE-16-2*, 2016.

[22] Raffaele Cappelli, Dario Maio, Davide Maltoni, and Ali Erol. Synthetic fingerprint-image generation. *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, 3:471–474 vol.3, 2000.

[23] Yu Chen, Bin Ma, and Zhuo Ma. Biometric authentication under threat: Liveness detection hacking. In *Black Hat Conference*, 2019.

[24] Tarang Chugh, Kai Cao, and Anil K Jain. Fingerprint spoof buster: Use of minutiae-centered patches. *IEEE Transactions on Information Forensics and Security*, 13(9):2190–2202, 2018.

[25] Tarang Chugh and Anil K Jain. Fingerprint spoof detector generalization. *IEEE Transactions on Information Forensics and Security*, 16:42–55, 2020.

[26] Frank. Chaos computer club breaks apple touchid, 2013. https://www.ccc.de/en/updates/2013/ccc-breaks-apple-touchid.

[27] Luca Ghiani, David A Yambay, Valerio Mura, Gian Luca Marcialis, Fabio Roli, and Stephanie A Schuckers. Review of the fingerprint liveness detection (livdet) competition series: 2009 to 2015. *Image and Vision Computing*, 58:110–128, 2017.

[28] Mordor Intelligence. Consumer biometrics market - growth, trends, covid-19 impact, and forecasts (2021 - 2026). Technical report, Mordor Intelligence, 2021.

[29] ISO/IEC. *ISO/IEC 30107-1:2016 information technology: biometric presentation attack detection - part 1: framework*. ISO/IEC, 2016.

[30] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar. *Handbook of Fingerprint Recognition*. Handbook of Fingerprint Recognition, 2009.

[31] Anthony J Mansfield and James L Wayman. Best practices in testing and reporting performance of biometric devices. 2002.

[32] Tsutomu Matsumoto, Hiroyuki Matsumoto, Koji Yamada, and Satoshi Hoshino. Impact of artificial "gummy" fingers on fingerprint systems. In *Optical Security and Counterfeit Deterrence Techniques IV*, volume 4677, pages 275–289. International Society for Optics and Photonics, 2002.

[33] Yaseen Moolla, Luke Darlow, Ameeth Sharma, Ann Singh, and Johan Van Der Merwe. Optical coherence tomography for fingerprint presentation attack detection. In *Handbook of Biometric Anti-Spoofing*, pages 49–70. Springer, 2019.

[34] Rodrigo Frassetto Nogueira, Roberto de Alencar Lotufo, and Rubens Campos Machado. Fingerprint liveness detection using convolutional neural networks. *IEEE transactions on information forensics and security*, 11(6):1206–1213, 2016.

[35] Unsang Park, Sharath Pankanti, and Anil K Jain. Fingerprint verification using sift features. In *Biometric Technology for Human Identification V*, volume 6944, page 69440K. International Society for Optics and Photonics, 2008.

[36] Paul Rascagneres and Vitor Ventura. Fingerprint cloning: Myth or reality, 2020. https://blog.talosintelligence.com/2020/04/fingerprint-research.html.

[37] Eric Setterberg. Before the ink is dry: Correcting biometric spoofing myths, 2020.

[38] Jonathan D Stosz and Lisa A Alyea. Automated system for fingerprint authentication using pores and ridge structure. In *Automatic systems for the identification and inspection of humans*, volume 2277, pages 210–223. International Society for Optics and Photonics, 1994.

[39] Bozhao Tan and Stephanie Schuckers. Spoofing protection for fingerprint scanner by fusing ridge signal and valley noise. *Pattern Recognition*, 43(8):2845–2857, 2010.

[40] Yulong Zhang, Zhaonfeng Chen, Hui Xue, and Tao Wei. Fingerprints on mobile devices: Abusing and leaking. In *Black Hat Conference*, 2015.

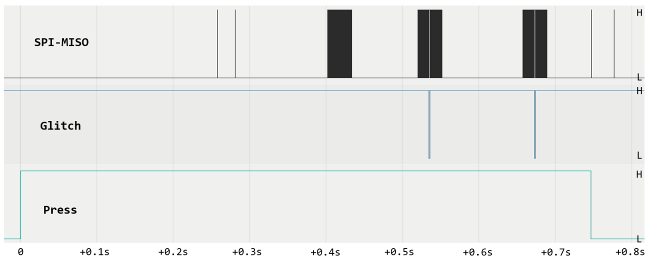


Figure 14: Signals captured during our CAMF attack on Touch ID within one attempt.

A CAMF on Touch ID

As mentioned in Section 8.2, we find that the attempt limit of Touch ID can not enlarge to infinite. This section explains why and how.

To trigger error-cancel in CAMF exploitation, we use the E1 type error discussed in Section 8.3. Figure 14 shows a malicious attempt under CAMF. The *Glitch* signal in the figure indicates whether the connection between the fingerprint sensor and processor is valid or not.

Unlike most Android devices, only three consecutive CAMF attempts can be exploited in Touch ID. We guess that Touch ID is equipped with a counter relevant to error-cancel, which may serve to prevent ghost touches or minor hardware failures. Nevertheless, we successfully make 15 actual attempts, which is three times as much as the attempt limit supposes. The whole process is described in two steps: (a) repeat the CAMF attempt twice when the screen is sleeping; (b) make normal attempts till a UI response is given when the screen is waking. A successful authentication result in any of the two steps can break the process. The second step ensures that the following matched fingerprint can unlock the device. Adversaries can repeat these steps until reaching the attempt limit.