JFCGuard: Detecting juice filming charging attack via processor usage analysis on smartphones

Meng, Weizhi; Jiang, Lijun; Wang, Yu; Li, Jin; Zhang, Jun; Xiang, Yang

Published in:
Computers & Security

Link to article, DOI:
10.1016/j.cose.2017.11.012

Publication date:
2018

Document Version
Peer reviewed version

Citation (APA):
Meng, W., Jiang, L., Wang, Y., Li, J., Zhang, J., & Xiang, Y. (2018). JFCGuard: Detecting juice filming charging attack via processor usage analysis on smartphones. *Computers & Security*, *76*, 252-264. https://doi.org/10.1016/j.cose.2017.11.012

Accepted Manuscript

Title: JFCGuard: Detecting juice filming charging attack via processor usage analysis on smartphones

Author: Weizhi Meng, Lijun Jiang, Yu Wang, Jin Li, Jun Zhang, Yang Xiang

PII: S0167-4048(17)30249-3
DOI: https://doi.org/10.1016/j.cose.2017.11.012
Reference: COSE 1239

To appear in: *Computers & Security*

JFCGuard: Detecting juice filming charging attack via processor usage analysis on smartphones

Weizhi Meng[1,2]*, Lijun Jiang[1,3], Yu Wang[1]**, Jin Li[1], Jun Zhang[4], and Yang Xiang[4]

[1] School of Computer Science, Guangzhou University, China

[2] Department of Applied Mathematics and Computer Science, Technical University of Denmark, Denmark

[3] Department of Computer Science, City University of Hong Kong, Hong Kong

[4]Faculty of Science, Engineering & Technology, Swinburne University of Technology, Australia

weme@dtu.dk;lijun.jiang@my.cityu.edu.hk;yuwang@gzhu.edu.cn;lijin@gzhu.edu.cn;junzhang@swin.edu.au;yxiang@swin.edu.au

Weizhi Meng is currently an assistant professor in the Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), Kongens Lyngby, Denmark. He received his B.Eng. degree in Computer Science from the Nanjing University of Posts and Telecommunications, China and obtained his Ph.D. degree in Computer Science from the City University of Hong Kong (CityU), Hong Kong in 2013. He was known as Yuxin Meng and prior to joining DTU, he worked as a research scientist in Infocomm Security (ICS) Department, Institute for Infocomm Research, Singapore, and as a senior research associate in CityU after graduation. He won the Outstanding Academic Performance Award during his doctoral study and won the HKIE Outstanding Paper Award for Young Engineers/Researchers in both 2014 and 2017. He is also a co-recipient of the Best Student Paper Award from the 10th International Conference on Network and System Security (NSS) in 2016. He is a member of Association for Computing Machinery (ACM) and IEEE. His primary research interests are cyber security and intelligent technology in security including intrusion detection, mobile security, biometric authentication, HCI security, cloud security, trust computation, and vulnerability analysis. He also shows a strong interest in applied cryptography.

*The author was previously known as Yuxin Meng, and the work was finalized during a visit at Guangzhou University.

**Corresponding Author: Email - yuwang@gzhu.edu.cn, phone and fax: 020-39366375

Lijun Jiang is a master student in the Department of Computer Science, City University of Hong Kong. He has actively participated in many research projects with board research interests including network security, intrusion detection, spam detection, and HCI security.

Yu Wang received his Ph.D. degree in computer science from Deakin University, Victoria, Australia. He is currently an associate professor with the School of Computer Science, Guangzhou University, China. His research interests include network traffic analysis, mobile networks, social networks, and cyber security.

Jin Li received his B.S. degree (2002) in Mathematics from Southwest University and the Ph.D. degree in Information Security from Sun Yat-sen University in 2007. Currently, he is a professor at Guangzhou University. He has been selected as one of the science and technology new stars in Guangdong province. His research interests include security in cloud computing and applied cryptography. He has published over 80 research papers in refereed international conferences and journals, and has served as the Program Chair or Program Committee Member in many international conferences.

Jun Zhang leads a research and development team working in cyber security. He received his PhD from University of Wollongong, Australia, in 2011. Jun Zhang is currently an associate professor in School of Software and Electrical Engineering, Swinburne University of Technology. He is also the deputy director of Swinburne Cybersecurity Lab (SCLab). He has published more than 80 research papers in refereed international journals and conferences, such as IEEE/ACM Transactions on Networking (ToN), IEEE Transactions on Image Processing (TIP), IEEE Transactions on Parallel and Distributed Systems (TPDS), IEEE Transactions on Dependable and Secure Computing (TDSC), IEEE Transactions on Information Forensics and Security (TIFS), and IEEE Transactions on Systems, Man and Cybernetics - Part B (TSMC-B), IEEE Transactions on Network and Service Management (TNSM). Jun Zhang received 2009 Chinese government award for outstanding student abroad. He is a member of the IEEE.

Yang Xiang received his PhD in Computer Science from Deakin University, Australia. He is the Dean of Digital Research & Innovation Capability Platform, Swinburne University of Technology, Australia. His research interests include cyber security, which covers network and system security, data analytics, distributed systems, and networking. In particular, he is currently leading his team developing active defense systems against large-scale distributed network attacks. He is the Chief Investigator of several projects in network and system security, funded by the Australian Research Council (ARC). He has published more than 200 research papers in many international journals and conferences, such as IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Information Security and Forensics, and IEEE Journal on Selected Areas in Communications. Two of his papers were selected as the featured articles in the April 2009 and the July 2013 issues of IEEE Transactions on Parallel and Distributed Systems. Two of his papers were selected as the featured articles in the Jul/Aug 2014 and the Nov/Dec 2014 issues of IEEE Transactions on Dependable and Secure Computing. He has published two books, Software Similarity and Classification (Springer) and Dynamic and Advanced Data Mining for Progressing Technological Development (IGI-Global). He has served as the Program/General Chair for many international conferences such as SocialSec 15, IEEE DASC 15/14, IEEE UbiSafe 15/14, IEEE TrustCom 13, ICA3PP 12/11, IEEE/IFIP EUC 11, IEEE TrustCom 13/11, IEEE HPCC 10/09, IEEE ICPADS 08, NSS 11/10/09/08/07. He has been the PC member for more than 80 international conferences in distributed systems, networking, and security. He served as the Associate Editor of IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, Security and Communication Networks (Wiley), and the Editor of Journal of Network and Computer Applications. He is the Coordinator, Asia for IEEE Computer Society Technical Committee on Distributed Processing (TCDP). He is a Senior Member of the IEEE.

Highlights

- We conduct a new study to explore the impact of JFC attack on both CPU and GPU usage,
- There is a more noticeable usage change by considering both CPU and GPU usage than considering CPU usage alone.
- We design JFCGuard, a security mechanism to intelligently detect JFC attacks for smartphone users.
- In the evaluation, we perform a new user study with over 250 participants to investigate the performance of JFCGuard.

Abstract. Smartphones have become necessities in people' lives, so that many more public charging stations are under deployment worldwide to meet the increasing demand of phone charging (i.e., in airports, subways, shops, etc). However, this situation may expose a hole for cyber-criminals to launch various attacks especially charging attacks and threaten user's privacy.

As an example, juice filming charging (JFC) attack is able to steal users' sensitive and private information from both Android OS and iOS devices, through automatically recording phone-screen and monitoring users' inputs during the whole charging period. More importantly, this attack does not need any permission or installing any pieces of apps on user's side. The rationale is that users' information can be leaked through a standard micro USB connector that employs the Mobile High-Definition Link (MHL) standard. Motivated by the potential damage of JFC attack, in this work, we investigate the impact of JFC attack on processor usage including both CPU- and GPU-usage. It is found that JFC attack would cause a noticeable usage increase when connecting the phone to the JFC charger. Then, we design a security mechanism, called JFCGuard, to detect JFC attack based on processor usage analysis for smartphone users. In the evaluation, we perform a user study with over 250 participants and the results demonstrate that JFCGuard can identify JFC attack in an effective way. Our work aims to complement existing research results and stimulate more research in this area.

Keywords: Charging Threat, Mobile Security, Smartphone Privacy, Android and iOS Device, Juice Filming Charging Attack, Detection Mechanism.

1 Introduction

Due to the widely adoption by millions of people, smartphones remain the most used handheld devices. International Data Corporation (IDS) reported that vendors shipped a total of 347.4 million smartphones worldwide in the first quarter of 2017 with an increase rate of 4.3%, which was slightly higher than the previous forecast of 3.6% growth [8]. As smartphones can offer a variety of applications for communication and entertainment, more and more users are likely to store their personal data on the phones, and use their phones for sensitive operations and transactions [28, 36]. As a result, people are often constantly using smartphones in their daily lives, which would greatly increase the demand of recharging their mobile devices and deploying public charging stations. As an example, Singapore Power (SP) promised to provide 200 free public charging stations for people to charge their mobile devices [38].

The free public charging stations are often available in public areas such as airports, shopping malls, and subways; however, these stations could expose a big hole for cyber-criminals to launch specific attacks, especially charging attacks, on threatening smartphone privacy and security. An early charging attack was designed by Lau *et al.* [13], where they presented Mactans, a malicious charger to launch charging attacks by injecting malware on an iOS6 device based on BeagleBoard during the charging process. Spolaor *et al.* [37] then proposed PowerSnitch, a malicious application that can utilize power consumption to send out data over a USB charging cable to the public charging station. Since we are not sure that these charging facilities are not maliciously controlled by cyber-criminals including charging station developers, maintenance managers and Government agencies, there is a need to pay particular attention on charging vulnerabilities in public charging facilitates.

Mactans and PowerSnitch highlighted the impact of charging attacks, but they could be only effective on either iOS or Android devices. In recent literature, Meng *et al.* [20] developed juice filming charging (JFC) attack, which is able to steal users' sensitive information on both Android OS and iOS devices, through automatically recording phone screen information during the charging process. All the interactions could be recorded as long as people keep charging and interacting with their phones. JFC attack is believed to be more scalable than Mactans and PowerSnitch, which can cause a wider impact on smartphone privacy. In summary, JFC attack can offer seven features: 1) can be easy to implement but quite efficient; 2) with less user awareness; 3) does not need to install any additional apps or components on phones; 4) does not need to ask for any permissions; 5) cannot be detected by any current anti-malware software; 6) can be scalable and effective on both Android OS and iOS devices; and 7) can automatically extract textual information from the collected videos by integrating OCR technology [21].

Due to the potential damage of charging attacks, there is a great need to identify such threat in an instant way. As JFC attack does not need to install any piece or require any permission on smartphone's side, it could have a large impact on users' privacy and would be more difficult to be detected than Mactans and PowerSnitch. Therefore, the motivation of this work is to design an appropriate security mechanism to detect JFC attack.

Contributions. Understanding how applications consume energy during execution is a promising method for threat detection, e.g., malware detection [10, 33]. In our previous work [1], it is found that JFC attack could cause an anomaly in CPU usage, which was however hard to raise the awareness of common users. Motivated by this, in this work, we focus on the detection of JFC attack and investigate its impact on processor usage including both CPU and GPU on smartphones. Then, we design JFCGuard, a security mechanism to identify JFC attacks for smartphone users. To our knowledge, this is an early work to discuss the detection of JFC attack via processor usage analysis. Our effort attempts to complement existing results and stimulate more attention on charging attacks. The main contributions of our work can be summarized as below.

– Intuitively, it is hard to identify JFC attack via power consumption, as this attack occurs when the phone is connecting to a charger. It would not be more feasible to detect charging attacks through analyzing processor usage. In this work, we first conduct a study to analyze processor usage including both CPU and GPU on smartphones during JFC attack with ten mobile devices. It is found that JFC attack may cause a noticeable usage change when connecting the phones to the JFC charger.

– Since the processor usage may be affected by many applications, we further design JFCGuard, a security mechanism to intelligently detect JFC attacks for smartphone users. This mechanism employs a machine learning classifier to decide whether the current charging trial is a potential JFC attack. In the evaluation, we perform a study with over 250 participants to investigate the performance of JFCGuard. The results demonstrate that JFCGuard can promisingly detect JFC attack in an effective way.

Organization. In Section 2, we first introduce the background of juice filming charging (JFC) attack including how to implement it in a real scenario, and then review related work on charging threats and several kinds of attacks in inferring users' private information, e..g, malware, side channels and physical access attacks. Section 3 conducts a study to analyze processor usage during JFC attack with a set of smartphones, and Section 4 describes JFCGuard, a security mechanism to detect JFC attack in an intelligent and effective way for smartphone users. Section 5 conducts an evaluation with over 250 participants to investigate the performance of JFCGuard. Finally, we make a discussion in Section 6 and conclude our work in Section 7.

2 Background and related work

2.1 Background of juice filming charging attack

JFC attack [20, 21] was developed based on the observation that no permissions would be asked and no compelling notification (the indicators are very small and last only few seconds) would be shown when connecting iPhones or Android phones to a projector, while the projector can automatically display the phone screen according to the Mobile High-Definition Link (MHL) standard. Taking advantage of these, JFC attack is able to automatically video-record phone screen including all users' inputs on screens via a standard micro USB connector during the whole charging period. This attack reveals that the display of phone screen can be leaked through a standard micro USB connector which employs the Mobile High-Definition Link (MHL) standard. For iPhones, the lighting connector should be used.

JFC attack can be realized by placing a VGA/USB interface between the phone and the back-end. Fig. 1 depicts the high-level implementation of JFC attack. Generally, when users charge their phones to the JFC charger, their phone screens can be video-captured in the back-end via the VGA/USB interface. Then, the captured videos can be stored and processed in the bank end. Similar to previous studies [20, 21], this work employs the same VGA/USB interface (called VGA2USB[5]), a full-featured VGA/RGB frame grabber, which helps send a digitized video signal from VGA to USB.

Fig. 2 describes the practical deployment of JFC attack using VGA2USB, where Fig. 2 (a) shows a deployment for iOS devices and Fig. 2 (b) shows a deployment for Android devices. Generally, the VGA/USB interface is connected to a computer, which can then automatically record phone screen when connecting the phone to the JFC charger. For instance, Fig. 2 (a) presents that the iPhone screen can be visible in the back-end. Hence all screen information could be captured and stored in the back-end, including users' inputs relevant to typed passwords, PIN code, email address and chatting logs, etc. It is worth noting that the back-end (e.g., computer) and other cables can be integrated into a smaller hardware (e.g., raspberry pi) or a power bank to reduce awareness.

Figure 3 (a) further shows how JFC attack monitors the phone screen and generates a corresponding video. According to [21], JFC charger can check whenever a device is connected to the back-end, read the real-time stream and write it into a video file. That is, when the JFC charger detects a phone is being charged, it would create a video file and record all phone screen information during the whole charging period. Figure 3 (b) shows a captured screen of a smartphone, in which the chatting messages were well-recorded.

Video analysis. To extract textual information from the collected videos, JFC attack first can check the charger periodically to decide when to start the recording process (all recorded data will be stored in a folder), and then use optical character recognition (OCR) techniques to process video frames. In the end, JFC attack merges the textual data and clean redundancy, i.e., removing duplicate words and sorting the remaining words in alphabetical order [21].

[5] http://www.epiphan.com/products/vga2usb/.

Features. The related research studies [20, 21] have showed that JFC attack can provide seven distinct features, if compared with software-based threat like malware: 1) can be easy to implement but quite efficient; 2) can achieve less user awareness; 3) does not need to install any additional applications or components on phones; 4) does not need to ask for any permissions from users; 5) cannot be detected by any current anti-virus software; 6) can be scalable and effective on both Android OS and iOS devices; and 7) can automatically process collected videos (i.e., extracting textual information, filtering and inferring the intended content). As compared to malicious applications (malware), JFC attack could cause a larger smartphone victims due to the following advantages:

– *No permission requirement.* JFC attack does not need any permissions from smartphone users, while most malicious applications still require at least key permissions from smartphones.

– *No need to install any pieces.* JFC attack does not need to install any pieces of applications on smartphones, whereas malicious applications have to install at least one small component on smartphones for functionality.

Under JFC attack, all phone screen information and interactions made between the users and their smartphones during the charging period would be captured. These videos are usually sensitive that contain users' private and sensitive information, so that cyber-criminals can extract the demanded information from the collected videos. As a result, JFC attack has a potential to become a big threat for smartphone privacy and security. There is a significant need to design proper security solutions to defend against such threat in an effective way.

2.2 Related work

Charging attacks. In literature, the first practical charging attack was proposed by Lau *et al.* [13], called Mactans, which could launch malware injection attacks using BeagleBoard (a functional mini-computer on an 8cm x 7.5cm board) during the charging process. Such attack can access to a significant amount of personal data without the user's permission and can install hidden malicious software on the device, as long as the device is unlocked. However, one big limitation is that users have to unlock their devices and install developer licenses in the beginning.

Spolaor *et al.* [37] focused on Android devices and described how an adversary could leverage a malicious charging station to exfiltrate smartphone data via a USB charging cable using power consumption. They further developed PowerSnitch, an application that could send out bits of data in the form of power bursts by manipulating the power consumption of the device's CPU. Their empirical results presented that this application could successfully decode a payload of 512 bits with a 0% Bit Error Ratio (BER). One limitation of this attack is requiring users to pre-install a small application on their phones.

Different from Mactans and PowerSnitch, Meng *et al.* [20] designed a new kind of charging attacks (called juice filming attacks) based on a standard USB connector and HDMI, which is able to steal users' private information through automatically video-capturing the phone screen. The efficiency relied on the observations that users were not aware of any risk when charging their phones in public places and that most users would interact with their phone during the charging period. This attack can be scalable and effective on both Android OS and iOS devices, and does not require any permissions from user's side. Later, they proposed JuiceCaster, an enhanced JFC attack by integrating OCR technology. JuiceCaster can automate JFC attack and is more effective in intelligently extracting sensitive information from the captured videos. They also advised a framework of collecting smartphone information in large-scale via cloud environments [22] and showed that such threat could be further enhanced by adopting advanced OCR techniques [24].

Smartphones have become a major target for cyber-criminals, so that charging attacks have the potential to become a big threat. Mactans and PowerSnitch can infer users' information from either iPhones or Android phones, but they need to request users' permission and install some parts of an application. JFC attack is powerful as it is scalable on both iOS and Android devices. In addition to charging attacks, there is a line of research reported how to infer users' private information and data through malware, accelerometer side channels and physical side channels.

Malicious applications (malware) on smartphones. Malicious applications are very common to be used to disrupt mobile operations, gather sensitive information, and gain access to private data [32]. Lin *et al.* [15] designed *Screenmilker,* an app that can detect the right moment to monitor the screen and pick up a user's password when the user is typing in real time. Xing *et al.* [41] conducted research on the Android updating mechanism and found Pileup flaws, through which a malicious app can strategically declare a set of privileges and attributes on a low-version operating system (OS) and wait until it is upgraded to escalate its privileges on the new system. By exploiting the Pileup vulnerabilities, the app can not only acquire a set of newly added system and signature permissions, but also determine their settings and it can further substitute for new system apps. Andriesse and Bos [3] introduced a code hiding approach for trigger-based malware, which can conceal malicious code inside spurious code fragments. Thus, it is invisible to disassemblers and static backdoor detectors.

Idrees et al. [11] applied ensemble methods for malware detection and developed PIndroid a Permissions and Intents based framework for detecting malware on Android phones. It showed an accuracy of 99.8% with 1,745 real smartphone applications. Similarly, Palumbo et al. [31] indicated that it was important for machine learning classifiers to be deployable in real scenarios, and implemented an ensemble approach for malware detection. They combined SVM with Atomic Naive Bayes classifiers, which could offer fast speed and reliability against data diversification. Their evaluation considered about one million samples with a model trained over a massive sample set of 120,000 samples. Nowadays, malware detection is a big challenge in various domains like Email security [19, 42].

Accelerometer side channel. Cai and Chen [6] presented a side channel, motion, on touch screen smartphones with only soft keyboards. They identified that when users type on the soft keyboard on the smartphone, especially when he/she holds the phone by hand rather than placing it on a fixed surface, the phone vibration on touchscreens are highly correlated to the keys being typed. In their evaluation, they showed that they were able to infer correctly more than 70% of the keys typed on a number-only soft keyboard on a smartphone. Marquardt *et al.* [17] also demonstrated that an application with access to accelerometer readings on a modern mobile phone can use such information to recover text entered on a nearby keyboard. They showed that through characterizing consecutive pairs of keypress events, as much as 80% of typed content can be recovered. Schlegel *et al.* [36] designed *Soundcomber,* a stealthy Trojan with innocuous permissions that can sense the context of its audible surroundings to target and automatically extract a small amount of targeted private information such as credit card and PIN numbers from both tone- and speech-based interaction with phone menu systems.

Han *et al.* [9] presented that accelerometer readings can be used to infer the trajectory and starting point of an individual who is driving, and pointed out that current smartphone operating systems allow any application to observe accelerometer readings without requiring special privileges. They further demonstrated that accelerometers can be used to locate a device owner to within a 200 meter radius of the true location. Then, Owusu *et al.* [29] described that accelerometer readings are very powerful which can be used to extract entire sequences of entered text on a smartphone. They showed how a background application can use the accelerometer as a side channel to spy on keystroke information during sensitive operations such as account login. They presented that they could successfully break 59 out of 99 passwords using only accelerometer measurements logged during text entry. Miluzzo *et al.* [27] presented *TapPrints,* a framework for inferring the location of taps on touchscreens using motion sensor data and showed that inferring English letters could be done with up to 90% and 80% accuracy. Several classical work can be referred to [5, 12, 16, 40, 45].

Physical side channel. Most of these attacks are based on oily residues left on the touchscreen and the screen reflection from nearby objects. For example, Aviv *et al.* [4] explored the feasibility of smudge attacks on touch screens for smartphones. They considered different lighting angles and light sources and the results indicated that the pattern could be partially identifiable in 92% and fully in 68% of the tested lighting and camera setups. Later, Zhang *et al.* [44] presented a fingerprint attack against tapped passwords via a keypad instead of graphical passwords. Their experiments on iPad, iPhone and Android phone showed that in most scenarios, the attack can reveal more than 50% of the passwords. For the screen reflection, Raguram *et al.* [34] showed that automated reconstruction of text typed on a mobile device’s virtual keyboard is possible via compromising reflections such as those of the phone in the user’s sunglasses. By means of the footage captured in realistic environments (e.g., on a bus), they showed that their approach was able to reconstruct fluent translations of recorded data in almost all of the test cases.

3 Processor usage analysis for JFC attack

As explained above, JFC attack does not need to install any pieces of applications on smartphone’s side as well as not need any permissions from users, thus, it is very difficult for current security mechanisms to detect. However, our previous study [1] showed that JFC would cause the change of CPU usage on smartphones, as it has to stream the screen information out of the phone. It was also found that JFC attack could cause CPU usage to increase in a range from 1% to 4% and from 0 to 2% for both Android OS and iOS devices, under the clean and normal environment, respectively, whereas users are not easy to pay attention to such anomaly.

For recent smartphone development, its CPU and GPU can have some overlaps when it comes to compute-intensive things such as video and audio processing. Fig. 4 describes a typical

ARM-based smartphone hardware architecture, in which a smartphone’s processors are very different from those adopted in a PC or laptop due to design constraints. Actually, current smartphone processor is either a separate ARM core or a DSP extension of the application processor from ARM core. Android OS is the first one that employed ARM core architecture, which could provide good performance with low power cost. Application processors can run user’s applications with associated instructions from the middleware and the OS, including audio and players, games, image processing, internet browser, etc. While it is worth noting that the majority of graphics intensive applications are usually run with the help from the GPU.

By understanding the processor architecture of smartphones, we notice that measuring both CPU- and GPU-usage may result in a more noticeable change than considering CPU usage alone. In this section, we focus on processor usage of both CPU and GPU and investigate the relevant usage change caused by JFC attack. Similar to [1], we pre-installed an application in the back-end of JFC charge to read both CPU- and GPU-usage on smartphones, before and after the phone being connected to the charger (e.g., SystemSens[6] for Android phones). Fig. 5 shows an application to read the CPU of an iPhone. As a result, the change of processor usage can be easily computed.

To analyze the processor usage, we record and compare the change of CPU- GPU- usage before and after connecting the phone to the JFC charger. We mainly consider two conditions: clean environment and normal environment.

– **Clean environment.** To avoid any influence, we clean and stop running any additional applications on smartphones.

– **Normal environment.** In this condition, users would install a set of mobile applications on their phones. As a study, we randomly installed 10 applications on these phones from the relevant official application stores.

The processor usage analysis was conducted in our lab environment, where a total of ten phones were tested including five Android phones and five iPhones due to the availability (similar to [1]). We repeated charging each phone for five times and recorded the corresponding usage change. The changes of processor usage are shown in Table 1 and Table 2 for the clean and normal environment, respectively.

[6] https://github.com/falaki/SystemSens.

Clean environment. Table 1 indicates that the processor usage could be increased when connecting the phone to the JFC charger, since the phone screen would be streaming into the back-end equipment. As compared to the results in [1] (e.g., from 1% to 4%), the usage change is still distinct for different phone types, but is more noticeable by considering both CPU and GPU. This change of processor usage offers a better opportunity to detect JFC attack. The main observations are discussed below.

– When charging the phones to the JFC charger, it is found that processor usage was generally increased. The changes could be in a range from 6% to 12% and from 6% to 14% for Android OS and iOS devices respectively. For example, Android Nexus One may have a 7-12 percent increase and iPhone 6 may have a 7-12 percent increase. Under our experimental settings, iPhones often have a bigger change of processor usage than that of Android phones.

– It is also found that different types of phones can result in a distinct usage change. For instance, HUAWEI phone had a bigger change of processor usage from 6% to 11% than Android Nexus One in a range from 7% to 12%. The processor usage of iPhone 6 (from 7% to 13%) is a bit larger than that of iPhone 6s (from 6% to 11%).

Normal environment. In this environment, our interest is to investigate the change of processor usage in a normal scenario, where users may install various mobile applications on their phones. Previous work [1] showed that installed applications could affect the change scope of processor usage due to their resource consumptions in the back-end of smartphones. Table 2 describes the corresponding changes of processor usage and the main observations are described as below.

– In this condition, it is found that the changes of processor usage could become smaller for both iPhones and Android phones. This observation is in line with the previous study [1]. For example, HTC One X9 had a change from 6% to 9% under the normal scenario, but had a change from 7% to 12% in a clean environment.

– Regarding different phones between iPhones and Android phones, the previous study [1] indicated that iPhones could have a bigger CPU usage change than Android phones. For example, iPhone 4 had a bigger change of CPU usage than Android Nexus One, Android Nexus 5 and HUAWEI P9 LITE; however, it is not true when considering GPU, i.e., iPhone 4 had a smaller change of processor usage (from 6% to 10%) than Android Nexus One from (from 5% to 10%).

– Regarding the phones with the same operating system, it is similar that different phones usually had a distinct change scope in a normal environment. For example, HUAWEI P9 LITE changed processor usage from 5% to 8%, HTC One X9 had a change ranged from 6% to 9%, iPhone 6s had a change of CPU usage ranged from 5% to 8%, and iPhone 6 had a change of CPU usage in a range from 6% to 10%.

These results indicate that by integrating GPU-usage, the change of processor usage could become more noticeable than the previous work [1]. It is the same that the change of processor usage in a normal scenario would become smaller than that in a clean environment. In our study, the change scope is reduced from [6%, 14%] to [5%, 10%]. As compared to considering only CPU-usage, our study shows that considering both CPU- and GPU-usage could provide a better chance for designing a security mechanism to identify JFC attack for smartphone users.

4 JFCGuard: a security mechanism against JFC attack

Our study shows that JFC attack could increase processor usage in a range from 6% to 14% and from 5% to 10% for both Android OS and iOS devices in a clean mobile environment and a normal scenario, respectively. This opens an opportunity to detect charging threats, but smartphone users often pay less attention to such anomaly in practice. To better protect users' privacy, there is a need to design an intelligent security mechanism to analyze processor usage in an effective way. Motivated by this, in this work, we design JFCGuard, a security mechanism to defend against JFC attack for smartphone users. The high-level architecture of JFCGuard is depicted in Fig. 6, which consists of three major components: *processor usage monitor*, *processor usage analyzer*, and *decision and reaction*.

– *Processor usage monitor.* This component is responsible for monitoring the phone and collecting data including CPU- and GPU-usage, when the phone is connecting to a charger.

– *Processor usage analyzer.* This component is used to analyze the change of processor usage and identify an anomaly by means of a machine learning classifier.

– *Decision and reaction.* This component is responsible for collecting the analyzer's results and making the final decision whether there is a potential JFC attack. Reactions can be implemented in this component, i.e., notifying users about the detection of JFC attack and requiring to stop charging.

Detection threshold. As a study, this work employs a K-nearest neighbor (shortly KNN) classifier to help identify an anomaly when connecting the phone to a charging facilitates. The selection is based on some merits [18]. First, this classifier can achieve a high classification accuracy. Second, this classifier could achieve a fast speed during both training phase and classification phase, which is a desirable property for a resource-limited platform like smartphones.

For detection, the KNN classifier should calculate the Euclidean distance, which represents the similarity between current usage change and the pre-computed clusters. The shorter the distance, the more similar they are. The calculation of the Euclidean distance can be described as below:

$$[Distance\ (P1, P2)]^2 = \sum_{0}^{N} (P1_i - P2_i)^2 \qquad (1)$$

$P1_i$ and $P2_i$ are the values of the *ith* attribute of points *P*1 and *P*2 respectively. In this work, we select both the minimum and maximum increased processor usage as two attributes. The generic steps of KNN classifier can be described as below:

1. For a new point *p*, calculating its Euclidean distance from other points;
2. Identifying the nearest point with the class;
3. Classifying the new point to the same class.

5 Evaluation

In this section, we conduct a user study to investigate the performance of JFCGuard in detecting JFC attack in various scenarios, and collect users’ feedback on smartphone malware and charging threats.

5.1 User study

To explore the effectiveness of JFCGuard, we perform a study by setting up a real JFC attack environment in three places: a university, a company and a healthcare center, respectively. A total of 286 participants attended this study and agreed to share the statistical data. All participants are volunteers and have no background of information security (i.e., none of them attended any course in relation to information security). The detailed information of participants related to age and occupation is summarized in Table 3.

Procedure and steps. Before the study, we had seek approval from each organization. Then, we set up the JFC charger in each environment according to Fig. 2. It is worth noting that only the back-end computer and a charging cable are visible to users, in which participants can utilize the cable to charge their phones. JFCGuard was run in the back-end to analyze processor usage and identify JFC attack (it is hard to directly install any applications on users' phones due to security and privacy concerns).

All participants were asked to charge their phones to a normal charger and the JFC charger for three times each. Then, all participants were given a set of questions about their charging behavior and awareness on smartphone threats. In the end, we explained the purpose of this study and seek approval from all participants to use their statistical data including processor usage profiles.

Results analysis. As participants should charge their phones three times, we could collect a total of 858 (286 × 3) trials for normal charging and JFC charging, respectively. All the trials from the normal charger can be treated as normal data while the trials from the JFC charger can be treated as malicious data. We denote true positive (TP) as the number of true positive instances predicted as positive, true negative (TN) as true negative instances predicted as negative, false positive (FP) as true positive instances predicted as negative and false negative (FN) as true negative instances predicted as positive. Moreover, we consider two major metrics to evaluate the classifier performance: *classification accuracy (CA)* and *hit rate (HR)*, which can be defined as follows:

$$CA = \frac{TP + TN}{TP + TN + FN + FP} \qquad (2)$$

$$HR = \frac{TP}{TP + FP} \qquad (3)$$

In addition, we select two supervised machine learning algorithms for performance comparison: back-propagation neural networks (BPNN) and decision tree (DT), according to their popularity. 1) BPNN was developed by Rumelhart *et al.* [35], in which the fundamental advances represented by the BPNN were the inclusion of a differentiable transfer function at each node of the network and the use of error back-propagation to modify the internal network weights after each training epoch. 2) DT is a tree-like model that classifies the given data items according to the values of its attributes. A node of a tree shows an attribute and its relevant edges. Each edge connects two nodes or a node and a leaf [43].

We ran the training and classification for ten times. The average classification accuracy and average hit rate are shown in Fig. 7. It is found that JFCGuard could achieve the best performance with a CA of around 0.97 and a HR of around 0.99, as compared with DT and

BPNN. Moreover, we tested JFCGuard on smartphones and identified that the time consumption of JFCGuard (KNN), BPNN and DT is 1.4, 10.2, and 2.9 seconds. These results demonstrate that KNN is an appropriate classifier that can be adopted by JFCGuard, and that JFCGuard is promising and effective in detecting JFC attack.

5.2 User feedback

Similar to the previous work [1], during our study, we also gave participants a set of questions to investigate their awareness on their charging habits and smartphone threats. Table 4 presents users' feedback about their charging behavior and their awareness on smartphone threats. The main observations are discussed as follows.

– *Malware threat.* The first and the second question attempt to investigate users' attitude towards smartphone malware. It is shown that up to 221 (77.3%) participants were able to recognize one kind of malware, and that 189 (66.1%) participants have already installed anti-virus software to protect their phones from malicious applications. These responses demonstrate that current users have paid much attention to smartphone malware, and are willing to employ several security mechanisms.

– *Charging behavior.* The next three questions are relevant to their charging behavior. It is found that most participants (204 out of 286) have the need to charge their phones in public areas like subways and shops. Up to 176 (61.5%) participants might choose to use a public charging station, and 211 (73.8%) participants were likely to interact with their phones during the charging period, i.e., playing games or chatting with their friends.

– *Charging threat.* The last two questions aims to investigate users's charging threat. It is found that 186 (65.0%) participants expressed their security concerns when charging the phones to a public charging station. This shows that more users start paying attention to charging threats, whereas up to 208 (72.7%) participants had no idea of any particular charging attack. This situation indicates that there is a great need to develop proper security mechanisms and educate users about charging threats.

Actually, most results in Table 4 are in line with the observations in [1]. On the whole, it is found that users could have a better knowledge on smartphone malware, but may still be unfamiliar with charging threat on smartphones. Subsequently, charging attacks like JFC attack could have a large potential to cause more victims than malicious application (malware) due to the lack of users' awareness and attention.

6 Discussion

User awareness on the change of processor usage. During the study, participants were invited to charge their phones to our deployed JFC charger. In the last step, all participants were asked to identify whether there is a usage change after connecting their phones to the charger. It is found that 91 (31.8%) participants were able to mention charging attack, while the others could not state the correct reason. As compared to the change scope in [1], there is a more noticeable usage change by considering both CPU and GPU, but users still have difficulty identifying such anomaly in an instant way. This because common smartphone users seldom pay attention to their phone usage. In other words, users are hard to notice a charging attack in a quick manner. They may found that the caused change of processor usage is normal, as the real CPU usage on smartphones would be affected by various mobile applications. As a result, there is a great need to deploy proper security mechanisms like JFCGuard in practice, with the purpose of protecting users' privacy.

Mitigation strategies. The root cause of JFC attack is due to that Android OS and iOS devices allow screen mirroring without explicit permission granting. It is very necessary to protect users' privacy by taking proper countermeasures to defend against such attack. In this work, we thus design JFCGuard, a security mechanism to detect JFC attack in an effective way. In real-world applications, JFCGuard can work with other potential mitigation strategies to offer a more thorough protection.

– A necessary mitigation is that smartphones should notify and ask users for permissions before output of the display. This strategy aims to increase user awareness on charging anomaly and let them decide whether to go ahead when a notification is pop-up.

– Another solution is to use a safe USB to present data leakage such as USB Condom [39]. This USB can prevent data exchange when the device is plugged into another device with a USB cable. It achieves this by cutting off the data pins in the USB cable and allowing only the power pins to connect through. However, this solution does not work for certain charging attacks like PowerSnitch [37], which can leak information via analyzing power consumption.

– The use of biometric features can also be a potential solution. For example, if we use a fingerprint-based unlocking mechanism instead of inputting PINs, then JFC attack cannot capture these biometric secrets [23]. The key idea here is not to input secrets from the touch-screen, while using voice or other biometrics to interact with the phone.

– Educating users is an important solution to reduce the risk of charging threat by increasing user awareness, since human beings are always the weakest point in a security mechanism [2, 14]. Our study shows that most users are not aware of charging threats; thus, there is a significant need to educate users.

Future challenges. This work validates that processor usage analysis can open an opportunity to detect charging threat, but actually common users are not easy to notice such anomaly in practice. As a result, this work proposes JFCGuard, an intelligent security mechanism to detect JFC attack through analyzing both CPU- and GPU-usage for smartphone users. Our study shows that JFCGuard can achieve good performance and can be effective in detecting JFC attack. However, there may exist many more complicated scenarios affecting the detection effectiveness.

– In most real-world scenarios, users often open and run various mobile applications on their smartphones (e.g., social networking apps, games), when charging their phones to a public charging station. This situation can result in a more complicated condition, in which the change of processor usage may also be caused by those running apps in addition to a potential charging attack (i.e., some applications can adjust their processor usage when the phone is being charged). This may increase the false rate and lower the effectiveness of JFCGuard.

– In current study, we consider that JFC charger would stream the phone screen into a back-end equipment for processing. However, JFC attack can also be launched without the need to handle the captured video in the back-end, but just store or send the recorded videos (i.e., using a power bank or sending out the recorded videos to cloud for processing, as shown in Fig. 8). In such scenario, the change of processor usage may be less noticeable, and there is a need to establish a sensitive energy-aware security mechanism [7, 25, 26]. This is an interesting topic as one of our future directions.

– In this work, we mainly considered a KNN classifier and achieved satisfied performance. In practical usage, we acknowledge that more advanced algorithms can be considered in our future work. However, there is a balance should be made between classification accuracy and increased load, i.e., advanced algorithms may require more steps and consume more CPU load.

– The charging attack developed in this work can steal information during the interactions between users and their phones. However, some particular attacks do not require such interactions. For instance, Ossmann and Osborn [30] pointed out that a root shell could be obtained over a USB Connector without using USB. They used Galaxy Nexus running CyanogenMod as a testbed and found that the console command gave a root shell if accessed with their own developed TTL UART.

Overall, this work shows that JFC attack would cause an increase of processor usage, but the anomaly is believed to be difficult for users to detect, as compared with most malicious applications. This is because malicious applications usually need minimum operations on phone’s side, i.e., increasing noticeable CPU load after installing part of pieces. It is also found that the change of processor usage would be reduced in a normal scenario due to various applications running on the smartphones. Through working with other potential solutions, there is a chance to develop more accurate and sensitive security mechanisms to identify JFC attacks and other charging threats.

7 Conclusion

With the increasing need of re-charging smartphones, public charging stations are widely available, which may open a potential hole for cyber-criminals to steal users’ sensitive and private data by launching charging attacks, like juice filming charging (JFC) attack that can steal users’ private information from both Android OS and iOS devices through automatically recording phone screen, including all users’ inputs during the whole charging period. In this work, we focus on the detection of JFC attack and investigate the impact of JFC attack by analyzing both CPU- and GPU-usage, i.e., the usage difference before and after connecting the phone to the JFC charger. Our first study showed that JFC charger could increase the processor usage in a range from 6% to 14% and from 5% to 10% for both Android OS and iOS devices, under the clean and normal environment, respectively. Then, we design JFCGuard, a security mechanism to detect JFC attack for smartphone users based on processor usage analysis. The second user study demonstrated that JFCGuard could achieve good performance and could be effective in detecting JFC attack. Our work aims to complement existing research and to stimulate more research efforts on charging threats.

Future work could include investigating the impact of JFC attack on processor usage in more complicated scenarios, i.e., exploring the influence of various applications running on smartphones. Future work could also include more diverse phones in the evaluation and designing a larger evaluation to validate the obtained results.

Acknowledgments. We would like to thank all participants for their hard work in the user study. This work was partially supported by National Natural Science Foundation of China (No. 61472091), Natural Science Foundation of Guangdong Province for Distinguished Young Scholars (2014A030306020), Science and Technology Planning Project of Guangdong Province, China (2015B010129015) and the Innovation Team Project of Guangdong Universities (No. 2015KCXTD014).

References

1. Jiang L., Meng W., Wang Y., Su C., Li J.: Exploring Energy Consumption of Juice Filming Charging Attack on Smartphones: A Pilot Study. In: Proceedings of the 11th International Conference on Network and System Security (NSS), pp. 199-213 (2017)

2. Allam, S., Flowerday, S.V., Flowerday, E.: Smartphone information security awareness: A victim of operational pressures. Computers & Security 42, pp. 56-65 (2014)

3. Andriesse, D., Bos, H.: Instruction-Level Steganography for Covert Trigger-Based Malware - (Extended Abstract). In: Proceedings of the 11th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA), pp. 41-50 (2014)

4. Aviv, A.J., Gibson, K., Mossop, E., Blaze, M., Smith, J.M.: Smudge attacks on smartphone touch screens. In: Proceedings of the 4th USENIX conference on Offensive Technologies, pp. 1–7. USENIX Association (August 2010)

5. Asonov, D., Agrawal, R.: Keyboard Acoustic Emanations. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 3-11 (2004)

6. Cai, L, Chen, H.: TouchLogger: inferring keystrokes on touch screen from smartphone motion. In: Proceedings of the 6th USENIX Conference on Hot Topics in Security (HotSec), pp. 1-6, Berkeley, CA,USA, USENIX Association (2011)

7. Curti, M., Merlo, A., Migliardi, M., Schiappacasse, S.: Towards energy-aware intrusion detection systems on mobile devices. In: Proceedings of the 2013 International Conference on High Performance Computing and Simulation (HPCS), pp. 289-296 (2013)

8. IDC. Worldwide Smartphone Market Gains Steam in the First Quarter of 2017 with Shipments up 4.3%, April 2017. [Online Available:] http://www.idc.com/getdoc.jsp?containerId=prUS42507917.

9. Han, J., Owusu, E., Nguyen, L., Perrig, A., Zhang, J.: ACComplice: Location inference using accelerometers on smartphones. In: Proceedings of the 4th International Conference on Communication Systems and Networks (COMSNETS), pp. 1-9, New York, NY, USA (2012)

10. Hoffmann, J., Neumann, S., Holz, T.: Mobile malware detection based on energy fingerprints a dead end? In: Stolfo, S.J., Stavrou, A., Wright, C.V. (eds.) RAID 2013. LNCS, vol. 8145, pp. 348-368. Springer, Heidelberg (2013)

11. Idrees, F., Rajarajan, M., Conti, M., Chen, T.M., Rahulamathavan, Y.: PIndroid: A novel Android malware detection system using ensemble learning methods. Computers & Security 68, pp. 36-46 (2017)

12. Kune, D.F., Kim, Y.: Timing attacks on PIN input devices. In: Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS), ACM, New York, USA, pp. 678-680 (2010)

13. Lau, B., Jang, Y., Song, C.: Mactans: Injecting malware into iOS devices via malicious chargers. Blackhat USA (2013)

14. Li, W., Jiang, L., Meng, W., Kwok, L.F.: Trust It or Not? An Empirical Study of Rating Mechanism and Its Impact on Smartphone Malware Propagation. In: Proceedings of the 10th IFIP WG 11.11 International Conference on Trust Management (IFIPTM), pp. 146-153 (2016)

15. Lin, C.-C., Li, H., Zhou, X., Wang, X.: Screenmilker: How to Milk Your Android Screen for Secrets. In: Proceedings of Annual Network and Distributed System Security Symposium (NDSS), pp. 1-10, (2014)

16. Liu, J., Zhong, L., Wickramasuriya, J., Vasudevan, V.: uWave: Accelerometer-Based Personalized Gesture Recognition and Its Applications. Pervasive and Mobile Computing 5(6), 657-675 (2009)

17. Marquardt, P, Verma, A., Carter, H., Traynor, P.: (sp)iPhone: Decoding Vibrations From Nearby Keyboards Using Mobile Phone Accelerometers. In: Proceedings of ACM Conference on Computer and Communications Security (CCS), pp. 551-562, ACM New York, NY, USA (2011)

18. Meng Y, Kwok L.F.: Adaptive False Alarm Filter Using Machine Learning in Intrusion Detection. In Proceedings of the 6th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), pp. 573-584 (2011)

19. Meng, Y, Li, W., Kwok, L.F.: Enhancing Email Classification Using Data Reduction and Disagreement-based Semi-Supervised Learning. In: Proceedings of the 2014 IEEE International Conference on Communications (ICC), pp. 622-627 (2014)

20. Meng, W., Lee, W.H., Murali, S.R., Krishnan, S.P.T.: Charging Me and I Know Your Secrets! Towards Juice Filming Attacks on Smartphones. In: Proceedings of the Cyber-Physical System Security Workshop (CPSS), in conjunction with AsiaCCS'15, ACM (2015)

21. Meng, W., Lee, W.H., Murali, S.R., Krishnan, S.P.T.: JuiceCaster: Towards Automatic Juice Filming Attacks on Smartphones. Journal of Network and Computer Applications 68, pp. 201-212 (2016)

22. Meng, W., Lee, W.H., Krishnan, S.P.T.: A Framework for Large-Scale Collection of Information from Smartphone Users based on Juice Filming Attacks. In: Proceedings of the Singapore Cyber Security R&D Conference (SG-CRC), pp. 99-106 (January 2016)

23. Meng, W., Li, W., Wong, D.S., Zhou, J.: TMGuard: A Touch Movement-based Security Mechanism for Screen Unlock Patterns on Smartphones. In: Proceedings of the 14th International Conference on Applied Cryptography and Network Security (ACNS), pp. 629-647 (2016)

24. Meng, W., Fei, F., Li, W., Au, M.H.: Harvesting Smartphone Privacy through Enhanced Juice Filming Charging Attacks. In: Proceedings of the 20th Information Security Conference (ISC), 2017.

25. Merlo, A., Migliardi, M., Fontanelli, P.: On energy-based profiling of malware in Android. In: Proceedings of the 2014 International Conference on High Performance Computing and Simulation (HPCS), pp. 535-542 (2014)

26. Merlo, A., Migliardi, M., Caviglione, L.: A survey on energy-aware security mechanisms. Pervasive and Mobile Computing 24, pp. 77-90 (2015)

27. Miluzzo, E., Varshavsky, A., Balakrishnan, S., Choudhury, R.R.: TapPrints: your finger taps have fingerprints. In: Proceedings of MobiSys, pp. 323-336, New York, NY, USA (2012)

28. Mylonas, A., Kastania, A., Gritzalis, D.: Delegate the smartphone user? Security awareness in smartphone platforms, Computers & Security 34, pp. 47-66 (2013)

29. Owusu, E., Han, J., Das, S., Perrig, A., Zhang, J.: ACCessory: password inference using accelerometers on smartphones. In: Proceedings of the 12th Workshop on Mobile Computing Systems & Applications (HotMobile), pp. 1-6, ACM New York, USA (2012)

30. Ossmann, M., Osborn, K.: Multiplexed Wired Attack Surfaces. Black Hat USA (2013) https://media.blackhat.com/us-13/US-13-Ossmann-Multiplexed-Wired-Attack-Surfaces-WP.pdf

31. Palumbo, P., Sayfullina, L., Komashinskiy, D., Eirola, E., Karhunen, J.: A pragmatic android malware detection procedure. Computers & Security 70, pp. 689-701 (2017)

32. Peng, S., Yu, S., Yang, A.: Smartphone malware and its propagation modeling: A survey. IEEE Communications Surveys and Tutorials 16(2), 925-941 (2014)

33. Polakis, I., Diamantaris, M., Petsas, T., Maggi, F., Ioannidis, S.: Powerslave: Analyzing the Energy Consumption of Mobile Antivirus Software. In: DIMVA 2015, pp. 165-184 (2015)

34. Raguram, R., White, A.M., Goswami, D., Monrose, F., Frahm, J.-M.: iSpy: automatic reconstruction of typed input from compromising reflections. In: Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS), pp. 527-536, ACM New York, USA (2011)

35. Rumelhart, D., Hinton, G., Williams, R.: Learning representations by back-propagating errors. Nature 323, pp. 533-536 (1986)

36. Schlegel, R., Zhang, K., Zhou, X., Intwala, M., Kapadia, A., Wang, X.: Soundcomber: A stealthy and context-aware sound trojan for smartphones. In: Proceedings of the 18th Annual Network and Distributed System Security Symposium (NDSS), pp. 17-33, San Diego, CA, USA (2011)

37. Spolaor, R., Abudahi, L., Moonsamy, V., Conti, M., Poovendran, R.: No Free Charge Theorem: A Covert Channel via USB Charging Cable on Mobile Devices. In: Proceedings of the 15th International Conference on Applied Cryptography and Network Security (ACNS), pp. 84-102 (2017)

38. Singapore Power to provide 200 free mobile phone charging stations for SG50. (July, 2015) [Online Available:] http://www.straitstimes.com/singapore/singapore-power-to-provide-200-free-mobile-phone-charging-stations-for-sg50

39. The Original USB Condom. http://int3.cc/products/usbcondoms.

40. Vuagnoux, M., Pasini, S.: Compromising electromagnetic emanations of wired and wireless keyboards. In: Proceedings of the 18th Conference on USENIX Security Symposium, USENIX Association, Berkeley, CA, USA, pp. 1-16 (2009)

41. Xing, L., Pan, X., Wang, R., Yuan, K., Wang, X.: Upgrading Your Android, Elevating My Malware: Privilege Escalation through Mobile OS Updating. In: Proceedings of the 2014 IEEE Symposium on Security and Privacy, Berkeley, CA, USA, pp. 393-408 (2014)

42. Wen, S., Zhou, W., Zhang, J., Xiang, Y., Zhou, W., Jia, W., Zou, C.C.: Modeling and Analysis on the Propagation Dynamics of Modern Email Malware. IEEE Transactions on Dependable and Secure Computing 11(4), pp. 361-374 (2014)

43. Yuan, Y., Shaw, M.J.: Induction of fuzzy decision trees. Fuzzy Sets and Systems 69(2), pp. 125-139 (1995)

44. Zhang, Y., Xia, P., Luo, J., Ling, Z., Liu, B., Fu, X.: Fingerprint attack against touch-enabled devices. In: Proceedings of the 2nd ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM), pp. 57-68, New York, NY, USA: ACM (2012)

45. Zhuang, L., Zhou, F., Tygar, J.D.: Keyboard Acoustic Emanations Revisited. ACM Transactions on Information and System Security 13(1), 1-26 (2009)

Fig. 1. The high-level implementation of juice filming charging attack.

Fig. 2. The practical deployment of JFC attack using a VGA/RGB frame grabber (VGA2USB) for (a) iOS devices and (b) Android devices.

Fig. 3. Under JFC attack: (a) captured video and (b) the captured phone screen.

Fig. 4. A typical ARM-based smartphone hardware architecture.

Fig. 5. CPU usage for an iPhone.

Fig. 6. The architecture of JFCGuard to defend against JFC attack.

Fig. 7. The average classification accuracy and average hit rate for JFCGuard, BPNN and DT.

Fig. 8. JFC charging attack via powerbank and cloud environment.

Table 1. Clean environment without the influence of other mobile applications: processor usage change when connecting the phone to the JFC charger.

Android OS	CPU Changes (%)	iOS	CPU Changes (%)
Android Nexus One	+ (7 ~ 12)	iPhone 4	+ (8 ~ 14)
Android Nexus 5	+ (7 ~ 10)	iPhone 5s	+ (7 ~ 10)
Samsung Note 5	+ (7 ~ 12)	Apple iPhone SE	+ (6 ~ 11)
HTC One X9	+ (7 ~ 12)	iPhone 6s	+ (6 ~ 11)
HUAWEI P9 LITE	+ (6 ~ 11)	iPhone 6	+ (7 ~ 13)

Table 2. Normal environment with other mobile applications: processor usage changes when connecting the phone to JFC charger.

Android OS	CPU Changes (%)	iOS	CPU Changes (%)
Android Nexus One	+ (5 ~ 10)	iPhone 4	+ (6 ~ 10)
Android Nexus 5	+ (6 ~ 10)	iPhone 5s	+ (6 ~ 9)
Samsung Note 5	+ (5 ~ 9)	Apple iPhone SE	+ (5 ~ 8)
HTC One X9	+ (6 ~ 9)	iPhone 6s	+ (5 ~ 8)
HUAWEI P9 LITE	+ (5 ~ 8)	iPhone 6	+ (6 ~ 10)

Table 3. Detailed information of participants in the user study.

Age Range	Male	Female	**Occupation**	Male	Female
18-25	101	94	Students	108	95
25-45	32	29	Personnel	33	31
Above 45	18	12	Senior People	10	9

Table 4. User feedback on malware threat, charging behavior and charging threat.

Questions	# of Yes	# of No
Do you install any anti-virus software on your smartphones?	189	97
Do you know any particular smartphone malware?	221	65
Do you have the need to charge your phone in public places such as shops, subways, airport, and so on?	204	82
Are you willing to use a public charging station (e.g., in airport, subways)?	176	110
Will you interact with your phone during charging (i.e., playing games, chatting with friends, etc.)?	211	75
Do you have any security concerns when charing the phones to a public charging station?	186	100
Do you know any particular charging attack?	78	208