Data Privacy and User Consent: An Experimental Study on Various Smartphones

Ayushi S Mehta, Vibha Dhiraj Puthran, Prasad Honnavalli
Department of Computer Science - ISFCR
PES University Bangalore, India

Abstract

A user's consent is required for sensitive information to be shared with third parties. Data exfiltration is one of the main end goals of cyber-attacks and with the dominance of Android in the smartphone market, many Android users are susceptible to this data theft. Most of these Android users are unaware of the theft of their personal data to an unauthorised entity. The entities responsible for this leakage can vary from malicious hackers and mobile network operators to the phone company themselves. Our paper aims to detect such data breach of privacy using two methods. First by installing a packet analyser computer program (tcpdump) directly on the phone and second by using a USB Adapter on nearby phones. Results show that data is being exfiltrated from the phones we tested on and phones from one country, in particular, are more susceptible to data exfiltration.

1. Introduction

A cyber-attack is defined as an attempt to disclose, modify or obtain illegal access to a computer system or any smart device. Confidentiality, integrity, authenticity and availability are known as the CIAA quartet and are the core security principles of Information Security. As a result of the attack, these core principles of the resource may be compromised [1]. Cyber-attacks can be broken down into two broad types namely passive and active attacks. A passive cyber-attack compromises confidentiality by attempting to either gain ac- cess into or make use of information from the system but does not affect the system resources. An active cyber-attack compromises integrity and availability by attempting to affect an operation or modify a system's resources. A few common types of cyberattacks include, Malware, Distributed Denial- of-service (DDoS) attack, Ransomware, Man-in-the-Middle, Phishing, Data Exfiltration and so on. This paper mainly focuses on Data exfiltration which is a type of a privacy breach that occurs when data is transferred without any permissions. It is most commonly performed by cyber-criminals over the Internet [2].

Data Exfiltration can by itself be the main attack or can be a very common consequence of many attacks. Successful attacks can cause a loss of sensitive customer data including personal information and debit/credit card numbers. This gives cyber criminals the power to sell the stolen personal details on the dark web, demand ransom, or harass one's customers. Hackers may also use this sensitive information for impersonation or identity theft. Not to mention, the impact of the breach can cause immense damage to the victim.

A smartphone contains personal as well as confidential business information. It records every activity of the user including their location at all times, their social relations, their interests, their personal pictures, their passwords and so on. Attacks on mobile devices are typically an easier target and give a bigger reward. The Google Android and Apple iOS jointly possess almost 99 percent of the global market share, but Android maintained its position as the leading mobile operating system worldwide, controlling the mobile OS market with a 74.13 percent share [6].

To secure data and codes of one app from other apps, Android uses a sandbox like environment. However, it also offers access to a central data repository which allows apps to share data as and when necessary making it vulnerable to data exfiltration [7]. Android has a policy of open-source kernel which opens doors for attackers to conduct their attacks [8]. In a recent study, researchers from International Computer Science Institute (ISCI) found that 1325 Android apps harvest user data despite being denied permission [9]. The motive of this paper is to detect data exfiltration on various smartphones using two methods. We tested five brands of phones which we shall label Brand A, Brand B and so on to maintain confidentiality. Brand A, B and D belong to one country whereas Brand C and E belong to another. All phones except Brand E are Android Phones.

Section II of our paper discusses the related work in this domain. Section III and IV explains the setup, approach and analysis of detecting data exfiltration for the two methods respectively. Section V states the final conclusion of our analysis and compares the two methods.

 | DOI: 10.20533/ijds.2040.2570.2021.0206

2. Related Work

Current data exfiltration detection methods for Android mostly focus on mobile applications leaking sensitive data. For example, in [3], a J48 classification algorithm was used to detect leakage of sensitive data by mobile applications and they also used the K-Means clustering algorithm to detect ma- licious mobile applications by comparing them to applications downloaded from Google Play Store. The proposed system can detect malign and benign applications.

In [4], a static taint analysis was used to track the flow of personal information in different Android applications to identify HTTP-based information exfiltration. The results revealed that multiple android applications are interested in identity- related information and the vast majority of them leak multiple types of sensitive information.

Similarly in [5], an anomaly-based approach is used for the detection of data exfiltration called passive application fingerprinting, where they passively extract fingerprints for each Android application. They also presented honey traffic, a deception-based detection system to identify mimicked communication. However, HTTPS traffic could be analyzed with these methods. Our method, on the other hand, is concentrated on detect- ing data exfiltration from an assortment of smartphones and studying the leaked information as well as tracing the route of the packets exfiltrated and analyzing the destination.

3. Detection via Tcpdump

3.1. Setup and Approach

In this method, we capture packets going out of the phone by installing tcpdump, a command-line interface that captures and analyzes network traffic [10]. We monitor the packets round the clock and filter analyze them via Wireshark. Figure. 1, shows the System Architecture for method 1. The re- quirements for this setup include a rooted Android smartphone with tcpdump installed and an active internet connection, a laptop that acts as our sniffer and a USB connector.

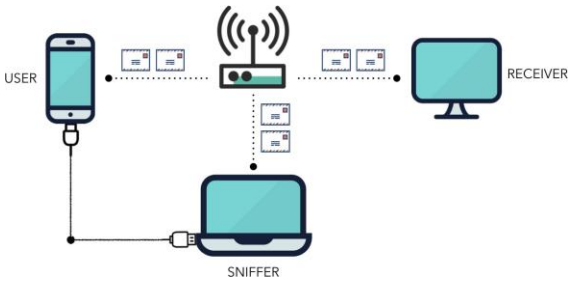

Figure 1. System Architecture for method 1

Three different brands of smartphones were tested in this method which we will refer to as Brand A, B and C. We fol- lowed a general methodology for all three phones but tailored steps for each phone accordingly. As the main objective of this method is to install tcpdump directly on the android phone and capture the packets, rooting is an essential first step to do so. Steps to root a phone varies for each model.

The first step to root any android phone is to unlock the bootloader. Every Android phone has a bootloader that instructs the operating system kernel to boot normally. A bootloader is usually locked on an Android device because although it is an open-source OS, the manufacturers want their customers to stick to their Android OS version specifically designed for that device. In order to enforce this concept, manufacturers lock the bootloader. With a locked bootloader, it is virtually impossible to flash a Custom ROM and forced attempts usually brick the phone. We could not root Brand A as the company had discontinued the service of letting users unlock the bootloader. As this phone could not be rooted, packets via tcpdump could not be captured. With Brand B, we first unlocked the bootloader. To do this, we enabled Developer Options, OEM unlocking and turned-on USB debugging. Then we were required to install the com- pany's official utility tool as it only authorises legitimate users with the unlocking permissions. After acquiring the same, we connected Brand B to the PC and booted it in Fastboot mode and successfully unlocked the bootloader. After which a TWRP Recovery image was downloaded for the device and this image was flashed on the phone using fastboot. Then we booted the phone into TWRP Recovery Mode and transferred the Magisk flashable zip file to the phone's internal storage. This Magisk image was flashed via the TWRP interface. To confirm this flash, we rebooted the system and successfully obtained root access. Root access essentially refers to the act of acquiring access to commands, system files, and folder locations that are usually locked off for the user [11, 12, 13]. In case of Brand C, Unlocking this phone's bootloader was comparatively easier to Brand B. To do this, we enabled Developer Options, OEM unlocking and turned-on USB debugging. We connected the phone to our PC via USB and launched an ADB terminal. Then we rebooted the phone into Bootloader Mode and critically unlocked the bootloader using a single line adb command. A critical unlock allows one to directly flash bootloader files and is the only unlock compatible for Brand C. The phone was then rebooted again, and the bootloader was unlocked.

Next, we downloaded Magisk Manager APK on the phone and patched the latest boot image file for Brand C in the app. We then pulled the patched boot image from the device to the PC via adb. And we reboot the phone into Bootloader Mode and flashed the patched boot image using an adb command. To confirm this flash, we restarted the system and

 | DOI: 10.20533/ijds.2040.2570.2021.0206

successfully rooted the phone. A summary of the steps for rooting all the three aforementioned phones is given in Table 1. After rooting the phones, we then downloaded tcpdump on the PC and pushed it onto the rooted phone via adb. Tcpdump is a command-line interface for analysing data- network packets. It aids the user to display TCP, IP and other packets being transferred over a network to which the device is attached. The environment for both the phones were set up in a similar manner to gather real-time information. We downloaded everyday apps such as Facebook etc. on the phones and used it for the same amount of time to ensure equal comparison on both devices. Thereafter, we monitored packets round the

Table 1. Steps to root an Android Phone

	Brand A	Brand B	Brand C
Enable OEM Unlock	✓	✓	✓
Enable USB Debugging	✓	✓	✓
Unlock the bootloader	✗	✓	✓
Install TWRP Recovery	✓	✓	✗
Flash Magisk	✓	✓	✓
Root Enabled	✗	✓	✓

clock for several days via tcpdump and analysed these pcap files on Wireshark. The size of almost all pcap files captured were about 450 MB (~500,000 packets). Hence, we filtered there levant packets by retaining only the packets that were sent from the phone and filtering out all ICMP packets and DNS queries to the local DNS servers.

3.2. Analysis

No packets were captured on Brand A as the phone could not be rooted. On analysing the filtered traffic for both Brand B and C, we found the following packets to be suspicious. In Brand B, we found packets going to a particular destination throughout the night. Figure. 2 shows a highlighted packet containing Application Data being sent from the phone's IP address to that destination IP.

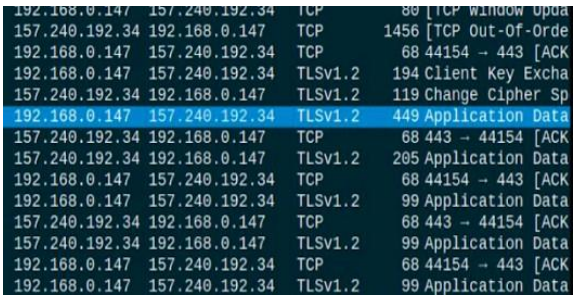

Figure 2. Packets going to facebook.com

On looking up this IP address on an online IP-Tracker, we found the destination to be servers of Facebook located in North America.

Since the packets were encrypted, we could not see what kind of data was leaving the phone. On some of the days we had intentionally logged out of Facebook from the phone but nevertheless, packets left the phone at spaced intervals between 11pm to 7am, when the phone was untouched.

Packets were also found going from the phone to graph.facebook.com, as seen in Figure. 3. Packets going to this destination were extremely frequent and were seen on all the days we captured the traffic. Facebook states that the 'Graph API' is the main way to get data in and out of its platform and uses this data to 'perform a wide variety of other tasks' which seems extremely suspicious [13].

In 2016, there was news about Brand B redirecting its user data from their US Amazon Web Services (AWS) and Singapore Servers to Chinese Servers.

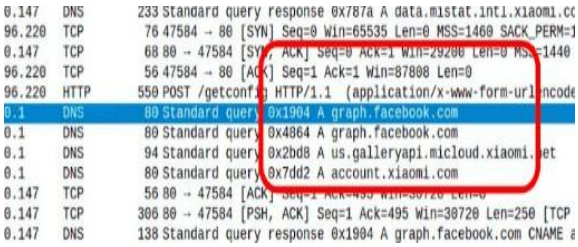

Figure 3. Packets going to graph.facebook.com

India then asked this brand to migrate all its Indian user data within the Indian subcontinent servers if the company wanted to continue its sales in the country. The company had decided upon AWS and subcontinent servers if the company wanted to continue its sales in the country. The company had decided upon AWS and Microsoft Azure infrastructure located within India to migrate its local data by the end of 2018. However, on analysing the captured packets, we found several packets still going to servers in the US and Singapore. Although we can't be sure of what kind of data is leaving the phone, the headquarters of these servers are located in China as can be seen in Figure. 4 and 5 respectively [14, 15].

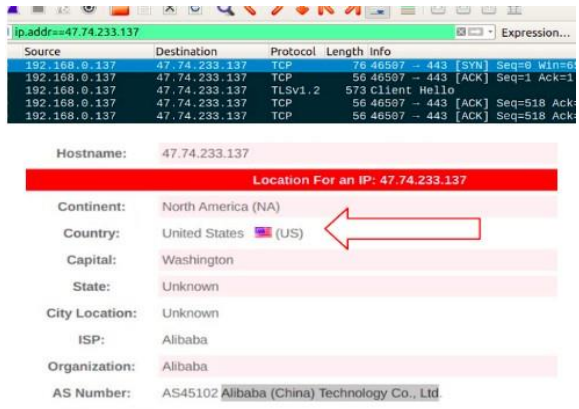

Figure 4. Packets going to US Servers

 | DOI: 10.20533/ijds.2040.2570.2021.0206

Data was also seen going to a server in Tokyo, Japan but unfortunately, we could not resolve who the organisation belonged to. Another bundle of packets was seen going to Brand B's servers for customisation of advertisements but there have been allegations stating that these servers were also holding personal information.

Most of the packets we found leaving Brand C were packets going to it's own servers. On setting up the phone, we were clearly prompted to give permissions to allow Brand C to take the phone's statistics regularly to provide a seamless and secure user experience. None of the packets examined by us

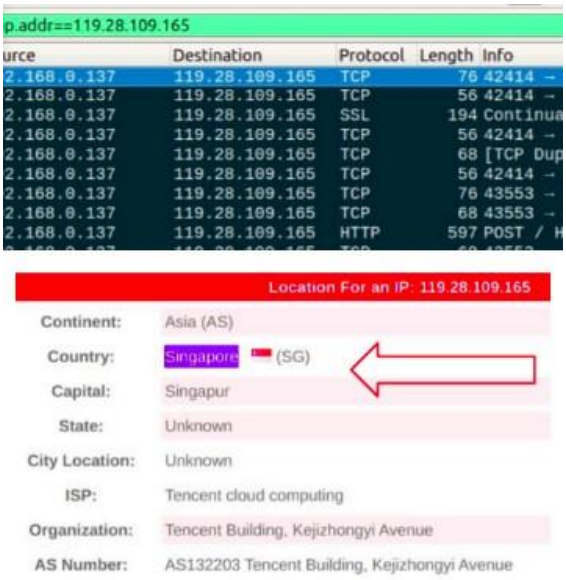

Figure 5. Packets going to Singapore Servers

seemed suspicious. We did find packets going to Instagram at 10:49 pm, but on further examining we realised that the conversation was initiated by Instagram itself as it was a notification. No other packets going to Instagram were found.

4. Detection via USB Adapter

4.1. Setup and Approach

In this method, a USB Adapter is required to sniff packets from nearby smartphones. Monitoring traffic in IEEE 802.11 networks is important to perceive the behaviour and traffic patterns of the network [16]. We used the Alfa AWUS036NHA Long Range USB Adapter which helped us to launch IEEE 802.11b/g/n wireless network in the 2.4GHz/5GHz band at around 100-200 Mbps [17].

The 802.11 is an over-the-air interface protocol between a wire- less user and a base station or between two wireless users. A USB Adapter overrides the computer's built in wireless card and hence receives signals from nearby routers [18]. It is essential to ensure that the chosen access point and the USB Adapter are compatible with each other by comparing the range and frequency. Figure. 6 shows the System Architecture for this method. The requirements for this setup include multiple Android smartphones, an access point, a laptop and an USB adapter.

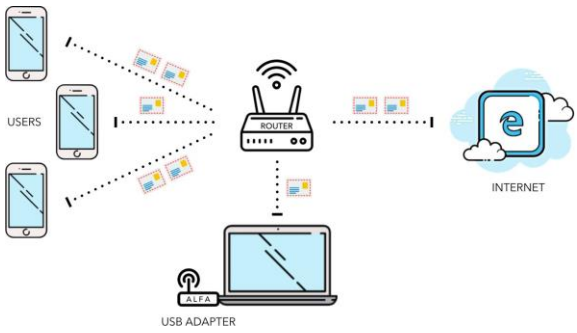

Figure 6. System Architecture for method 2

Five different brands of smartphones were tested in this method, which we will refer to as Brand A, B, C, D and E. The Alfa Adapter has various modes in which it works. For packet sniffing, the promiscuous and monitor modes are used. It is important to note that these two modes are not the same. Monitor mode does not require an association with any access point or ad-hoc network. and can only be used for wireless connections, whereas promiscuous mode needs to have an association and can be used for both wired and wireless connections. Thereby, in order to sniff packets of all the phones in a nearby vicinity that are connected on a wireless channel, we enabled monitor mode on the adapter using Aircrack which is a suite of tools for assessing the Wifi network security. We then set out to capture packets of the phones connected to a particular access point in our vicinity. If the access point is WPA protected, the traffic captured will be encrypted and mainly contain 802.11 packets. To decrypt this traffic, knowledge of the wifi password is essential which we attempted to crack using Aircrack tools.

First, we identify all Wifi access points in a particular range, as the USB adapter is already in monitor mode. We choose a victim from the list of access points and capture the encrypted 4-way handshake by carrying out a de-authentication attack. For a successful attack, we need to have the client authenticate against the access point. If they are already authenticated, we can de-authenticate them and their system will automatically re-authenticate, whereby we can grab their encrypted password in the process [20]. The air cracking command takes as input a file, composed of a list of passwords and runs this file on the encrypted 4-way handshake captured. A good side note to remember is that this type of attack is only as good as your password file. If the actual password is present in the list, the cracking will be

 | DOI: 10.20533/ijds.2040.2570.2021.0206

successful. Hence if a strong password is not used, it might be very easy for a hacker to crack.

We then added the cracked WPA decryption key on Wire- shark to decrypt the traffic and obtain DNS, TCP, HTTP packets. However, one limitation with this procedure was that only packets from one phone could be decrypted at a time.

To overcome this limitation, one can use open networks as it directly captures decrypted packets from multiple phones simultaneously. Almost all pcap files captured contained about 1,200,000 packets out of which each phone's packets were only 500 TCP packets. We again filtered relevant packets for each phone in a similar manner to method 1.

4.2. Analysis

In method 1, we did not capture any packets from Brand A as the phone could not be rooted. However, on analysing packets captured packets in method 2, we found data going to Facebook servers as well as Amazon servers. We also found bursts of packets going to a server in Singapore from 12am to 7am as shown in Figure 7. The Organization for this IP Address was a company called 'AK Steel

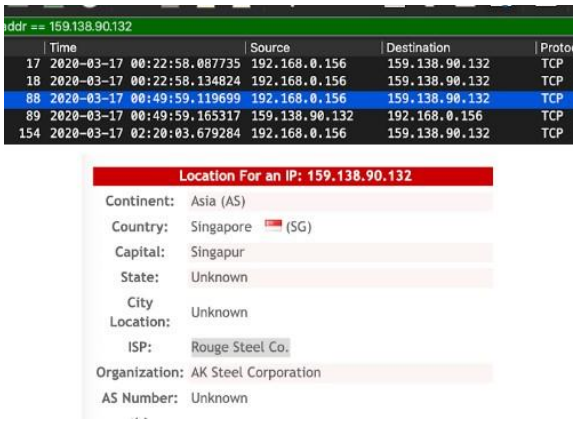

Figure 7. Packets going to AK Steel

We had surfed amazon.com on the browser during the day and on examining the packets captured that night, we found data going to amazon.com at 11:57pm as seen in Figure. 8. We were sceptical about this since we had never logged in to amazon from this phone and also made sure to close all the tabs as soon as we were done browsing.

Method 2 was the first time Brand D was being tested on. Analysing the captured packets, we again found data going to Facebook servers. Data was also seen going to Akamai Technologies and General Electric Company as shown in Figure. 9. Nevertheless, there might be a possibility of Brand D renting these companies' servers to store their data.

Packets were also found going to Unisys Corporation as shown in Figure. 10. In 2006, this company was being investi- gated by the FBI for allegedly transmitting data to Chinese Servers [19].

Figure 8. Packets going to Amazon.

Corporation' which is a steel making company headquartered in Ohio. For data to go throughout the night to an unrelated, random company definitely seems out of the ordinary.

For Brand B, we found packets going to Facebook servers which we have already seen during the analysis phase of Method 1. The reasons stated in the previous method stand true for these packets to be suspicious. Also, similar to method 1, data was found going to servers located in Singapore.

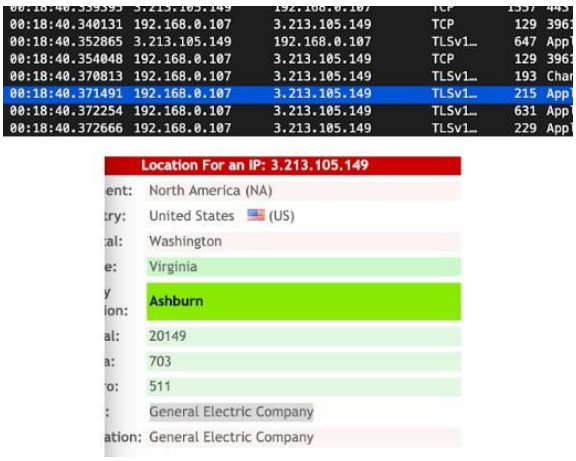

Figure 9. Packets going to General Electric Company

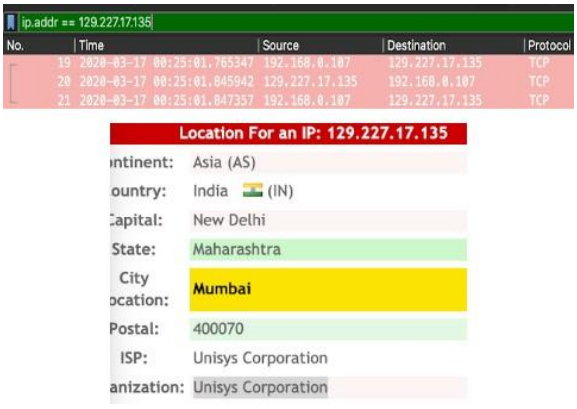

Figure 10. Packets going to Unisys Corporation

Data was found going to Verizon Business EdgeCast Networks located in Australia as seen in Figure. 11. Verizon is a network carrier collaborating with Brand D in the UK and it seems very dubious as Verizon has no connection with Brand D in India.

Figure 11. Packets going to Verizon Business EdgeCast Networks

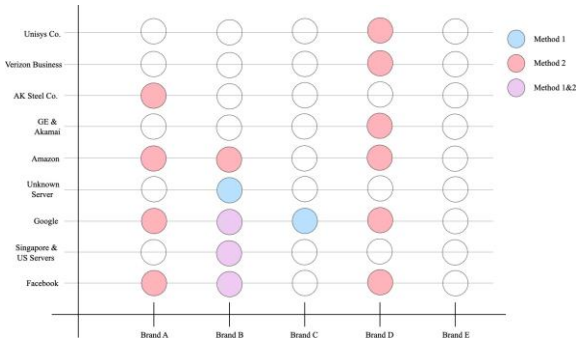

Figure 12. Summary of Results

Packets from both Brand C and Brand E could not be captured using Method 2. In case of an open network, both phones' IP Address was not visible on Wireshark and with a password protected network, even after entering the decryption keys, packets from neither phone got decrypted.

5. Conclusions and Future Work

To conclude, our initial assumption that smartphones from one country in particular are more susceptible to data exfiltration as compared to phones from other countries continues to stand true as can be seen by our analysis.

A list of all the suspicious websites / servers to which packets from each phone were being exfiltrated to has been summarised in Figure. 12. From our analysis, Brand A and Brand D seem more vulnerable to data exfiltration compared to the other phones we tested.

In method 1, we captured packets via tcpdump of one phone at a time and obtained 100,000 packets out of which 45,000 were TCP packet whereas in method 2, we captured packets of all the phones in that vicinity simultaneously via a USB Adapter and obtained 120,000 packets out of which only 500 TCP packets were captured per phone. Hence, although method 2 was comparatively easier to perform method 1 was more insightful. Whilst we successfully executed all the tasks that we initially planned, we would like to take a step further and attempt to decrypt the data to confirm our assumptions on the type of data being exfiltrated and perform the experiment on a vast variety of phones with more apps downloaded on them to give us a better analysis.

One of the major limitations that we faced in method 1 was not being able to root all the phones that we tested on pre- venting the installation of tcpdump on it. Hence, discovering a method to install tcpdump on an unrooted phone would result in a more insightful analysis.

6. References

[1] Abi Tyas Tunggal, 'What is a Cyber Attack?', https://www.upguard.com/blog/cyber-attack, (Access Date: 22 August, 2019).

[2] Nate Lord, 'What is Data Exfiltration?', https://digitalguardian.com/blog/what-data-exfiltration, (Access Date: 11 September, 2019).

[3] Y. Canbay, M. Ulker and S. Sagiroglu," Detection of mobile applications leaking sensitive data," 2017 5th International Symposium on Digital Forensic and Security (ISDFS), Tirgu Mures, 2017, pp. 1-5, doi: 10.1109/ISDFS.2017.7916515.

[4] Kelkar, Soham Kraus, Timothy Morgan, Daria Zhang, Jun-jie Dai, Rui. (2018). Analyzing HTTPBased Information Exfiltration of Malicious Android Applications. 1642-1 645. 10.1109/Trust- Com/Big DataSE.2018.00242.

[5] Bortolameotti, R. (2019). Detection and evaluation of data exfiltration. Enschede: University of Twente. https:// doi.org/10.39 90/1.9789036548243.

[6] S. O'Dea, 'Mobile operating systems' market share worldwide from January 2012 to December 2019', https: //www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/, (Access Date: 17 January, 2020).

[7] Praful Meshram. (2016). "A survey paper on vulnerabilities in Android OS and Security of Android Devices". Cummins College of Engineering for Women.

[8] Sravani Manukonda. (2018). "Understanding the Security Posture of Android Applications and OS". PES University, India.

[9] Android Apps Steal Your Data—Even If You Deny Permission, https://www.huffingtonpost.in/entry/android-apps-steal-your-data-even-if-you-deny-permissionin5d91c74ae4b0e9e7604f86b3, (Access Date: 01 August, 2019).

 | DOI: 10.20533/ijds.2040.2570.2021.0206

[10] Ricardo Gerardi, 'An introduction to using tcpdump at the Linux com - mand line', https://opensource.com/article /18/10/introduction-tcpdump, (Access Date: 10 November, 2019).

[11] Kelsey Rexroat and Paula Beaton, 'How to root Android phones and tablets (and unroot them)' https://www.digitaltrends.com/mobile/how-to-root-an dro id/, (Access Date: 11 February, 2020).

[12] Adam Sinicki, 'Root Android: Everything you need to know!', https://www.androidauthority.com/root-android-27 7350/, (Access Date: 21 January, 2020).

[13] Developers - Facebook, Overview- Graph API, https://developers.facebook.com/docs/graph- api/overview, (Access Date: 13 Febuary, 2020).

[14] Javed Anwer, 'Some Chinese phones caught sending personal data to Chinese servers', https://www.india today.in/technology/news/story/some-chinese-phones-caug ht-sending-personal-data-to-chinese-servers-352208-2016- 11-16, (Access Date: 16 March, 2020).

[15] Indianexpress, Xiaomi to migrate all of its local data to cloud servers located in India, https://indianexpres s.com/article/technology/tech-news- technology/xiaomi-to -migrate-all-of-its-local-data-to-cloud-servers-located-in-in dia-5333982/, (Access Date: 31 February, 2020).

[16] kirale, USB Dongle usage as packet sniffer, https://www.kirale.com/support/kb/usb-dongle-usage-as-pa cket-sniffer/, (Access Date: 3 March, 2020).

[17] Alfanetwork, Alfa Awus036nha AlfaNetwork India, https://alfanetwork.co.in/shop?olsPage=products%2Falfa-a wus036nhayoReviewsPage=1, (Access Date: 3 March, 2020).

[18] Webopedia, Vangie Beal, '802.11 IEEE wireless LAN standards, https://www.webopedia.com/TERM/8/802_11.h tml, (Access Date: 7 March, 2020).

[19] Wikipedia, Unisys', https://en.wikipedia.org/wiki/Unis ys, (Access Date: 15 March, 2020).

[20] Null-Byte, OccupyTheWeb, 'Cracking WPA2-PSK Passwords Using Aircrack-Ng', https://null-byte.wonderho wto.com/how-to/hack-wi-fi-cracking-wpa2-psk-passwords -using-aircrack-ng-0148366/, (Access Date: 3 March, 2020).

7. Acknowledgement

The authors would like to express gratitude to all faculty and staff of PES University and the Center for Information Security, Forensics and Cyber Resilience (ISFCR) for their continuous guidance, assistance and encouragement through- out the development of this research.

 | DOI: 10.20533/ijds.2040.2570.2021.0206