Analysis and Classification of Some Selected Media Apps Vulnerability

Olawale Surajudeen Adebayo[1](✉), Joel Sokoyebom Anyam[1], Shefiu Ganiyu[1], and Sule Ajiboye Salawu[2]

[1] Federal University of Technology Minna, Minna, Niger, Nigeria
{waleadebayo,shefiu.ganiyu}@futminna.edu.ng, anyamjoel01@gmail.com

[2] Aminu Saleh College of Education, Azare, Bauchi, Nigeria
salawu_sul@yahoo.com

Abstract. This research investigates popular messaging applications' traffic in other to assess the security or vulnerability of communication on those applications. The experiment was carried out in a Local Area Network. Wireshark, NetworkMiner and Netwitness Investigators were used to capture and analyse the traffic. Ten (10) instant messaging applications were installed on Android platforms and used for the experiment. Different types of sensitive media files were recovered from the network traffic, including images, documents/texts and audio. The Internet Service Provider (ISP) of the sender was also recovered along with the resident city of the third party. The research classifies the mobile applications into vulnerable and nonvulnerable applications using the gathered data. Thus, it was discovered that out of ten mobile applications investigated, only Viber application was non-vulnerable to tested attacks. The classification result also shows random forest as the best classifier using this research data.

Keywords: Social media · Vulnerability · Social media vulnerability · Instant messaging applications · Mobile applications · Vulnerability classification

1 Introduction

Instant messaging applications, running on portable devices such as smartphones and tablets, have become increasingly popular around the world [1]. As new messaging applications started to emerge and tried to replace traditional SMS, developing them with security and privacy in mind was not a top priority for the developers. Popular messaging tools used in recent years do not support end-to-end encryption, only standard client to server encryption, this gives the service providers access to more private information than required. The use of mobile applications for communication has experienced accelerated growth and has fast become the standard method of communication. Based on recent revelations of the widespread state surveillance of personal communication, many products now claim to offer secure and private messaging. However, these messaging applications have been proven over time to be vulnerable notwithstanding

S. Misra and B. Muhammad-Bello (Eds.): ICTA 2020, CCIS 1350, pp. 457–469, 2021.
https://doi.org/10.1007/978-3-030-69143-1_35

improvements. The leakage communication on instant messaging applications poses a serious risk especially in sensitive conversations like organizational matters, where sensitive data like a customer list or sales report may be revealed on the internet.

The efficiency and effectiveness of the information systems, in many ways, depend on its architecture and how data are transmitted among different parties. Similarly, a very crucial aspect of the software development is the security of data that flows through open communication channels. One of the most popular architecture is client/server architecture that makes the centralization of data storage and processing enable and provide flexibility for applying authentication methods and encryption algorithms within information systems. The increase in the number of clients necessitated the increase in the authentication and encryption level as high as possible. Client/server is a technology that allows opening an interactive session between the user's browser and the server [2]. One of the biggest potential security flaws for most authentication schemes in the client/server architecture is the ability of the users to ensure their secrecy [3]. In this work, authors perform an experimental analysis on ten instant messaging applications for the Android mobile phone operating system. The study reveals that despite the claim by instant messaging applications about providing end-to-end security, unencrypted data could still be retrieved from the traffic of these applications. The remainder of this paper is organized as follows: the related works on the analysis of instant messaging vulnerability was discussed in section two; the research methodology and experimental setup were discussed in section three. In section four, the research experiment and implementation were highlighted. The overview of the experimental results was discussed in section five while the conclusion and further research were provided in sections six and seven.

2 Related Works

Dickson and Messenger [4] carried out experiments to mimic the reality of an investigation linking one party to another, including 'pre-determined' text conversations and the exchange of data files. Monitoring software was used at all times on the suspect's system to record the changes being effected to it. The results show that conversation content and transferred files, evidence of contact, display picture, transmitted files, connection logging, unallocated clusters and swap file, network passwords and data on the RAM were retrieved. Meghanathan, Allam, and Moore [5] discusses different tools and techniques available to conduct network forensics. The tools discussed include: eMailTrackerPro to identify the physical location of an email sender; Web Historian to find the duration of each visit and the files uploaded and downloaded from the visited website; packet sniffers like Ethereal to capture and analyze the data exchanged among the different computers in the network. Default iPhone web browser Safari (customized for iPhone presented) was used by Husain and Sridhar [6] to test for three Volatile Instant Messaging (VIM) applications where participants can enjoy instant messaging by just using a web browser without installing any application on the user's local system. In their results, unique Phrases, timestamps, screen names and buddy list for Aim and Yahoo were recovered while Plain text passwords for AIM were retrieved.

Chin, Felt, Greenwood, and Wagner [7] while analysing inter-application communication on Android platforms conducted an Intent-Based Attack using ComDroid for vulnerability and bug detection. They analysed 20 applications and found 34 exploitable vulnerabilities; 12 of the 20 applications had at the least one vulnerability. Vidas et al. [8] after exploring special device boot modes and Android's partitioning schema, detail the composition of an Android bootable image and discuss the creation of such an image designed for forensic collection. Their major contribution is a general process for data collection of Android devices. Their results show that the use of the recovery booting provides a consistent, repeatable method of collecting numerous Android devices without "rooting" the device in normal operating mode. Zhang, He, Liu, and Bridges [9] investigated users' online activities including web browsing, chatting, online gaming, downloading, uploading and video watching, etc. through traffic analysis. A hierarchical classification system based on machine learning algorithms was implemented to discover what a user is doing on his/her computer. Results show that their system can differentiate online applications on the accuracy of about 80% in 5 s and over 90% accuracy if the eavesdropping lasts for 1 min. Appelman, Bosma, and Veerman [10] investigated how Viber performs security wise in comparison to other services. A definitive conclusion was not found but most details of the protocol used to transfer the voice data were documented. The application code was analysed but no real weaknesses were found. Results show that in the Local Storage everything is unencrypted and can be viewed fairly easily, although your phone needs to be rooted or jailbreaked while for the Transferred Data, Manual Reverse Engineering is possible but is a long process which requires a lot of experience.

Schrittwieser et al. [11] analysed nine popular mobile messaging and VoIP applications and evaluated their security models with a focus on authentication mechanisms. They found out that most of the examined applications use the user's phone number as a unique token to identify accounts, which further restricts the implementation of security measures. Results show that major security flaws exist in most of the tested applications, allowing attackers to hijack accounts, spoofsender-IDs or enumerate subscribers. Adami, Callegari, Giordano, Pagano, and Pepe [12] proposed a real-time algorithm (named Skype-Hunter) to detect and classify Skype traffic. By means of both signature-based and statistical procedures, they were able to correctly reveal and classify the signalling traffic as well as the data traffic (calls and file transfers). Thakur [13] conducted a Forensic Analysis of WhatsApp on Android Smartphones in which they carry out a live analysis of an Android smartphone to extract user interaction information from the whatsApp application's volatile and non-volatile memory. The results for the volatile memory show that critical application data is present in the RAM and can be extracted for further analysis, while for the Non-volatile memory, all similar applications that load data from a SQLite database can be tested for data recovery using non-volatile memory forensics. Mahajan and Sanghvi [14] used Cellebrite UFED (Universal Forensic Extraction Device) Classic Ultimate (V 1.8.0.0), to extract files and folders from five (5) android devices. Rafique, and Khan [15] present a critical review of static and live analysis approaches and evaluate the reliability of different tools and techniques used in static and live digital forensic analysis.

Coull and Dyer [16] show that it is possible for an eavesdropper to learn information about user actions, the language of messages, and even the length of those messages with greater than 96% accuracy despite the use of state-of-the-art encryption technologies simply by observing the sizes of encrypted packets. Anglano [17] used a set of YouWave virtual machines, namely one for each device involved in the experiments, running Android v. 4.0.4. On each one of these machines WhatsApp Messenger v. 2.11 was installed and used. SqliteMan was used to examine the databases maintained by WhatsApp Messenger and notepad++ to examine textual files. The results indicate that list of contacts, inference also of when a specific contact has been added, recovery of deleted contacts, time of deletion, deleted messages, time of deletion and users that exchanged them. Karpisek, Baggili, and Breitinger, [18] were able to decrypt the network traffic and obtain forensic artifacts that relate to this new calling feature on WhatsApp, which included the WhatsApp phone numbers, server IPs, audio codec (Opus), call duration, and call termination using Wireshark v1.12.5, 32-bit, with the WhatsApp dissector and Pidgin v2.12.115, 32-bit, with the WhatsApp plugin. [11] Abdul Aziz, Mokhti, Nadhar, and Nozri [19] studied and experimented several techniques on the extraction and analysis of smartphones' data using Sleuth Kit Autopsy. Authors' aim was to identify methods of extracting and analysing data on android based smartphone. Authors were able to extract email and contact artefacts. Walnycky, Baggili, Marrington, Moore, and Breitinger [20] acquired the traffic capture files and conducted examination using Wireshark, NetworkMiner, and NetWitness Investigator, to extract Text chat, Audio, video, image, sketch, and location sharing. This research follows this methodology given its high level of efficiency in exposing vulnerabilities as shown in past material. We worked on a wide range of updated applications with a focus on text/documents, audio, ISP, image, and location sharing.

Adebayo, Sulaiman, Osho, Abdulhamid, and Alhassan [21] in their seminar paper focus on the forensic analysis of Kik messenger which is a multi-platform instant messaging application on android devices. Authors captured and examined data related Kik and forensic images of three android devices with android versions 4.4 (KitKat), and 5.0 (Lollipop) and different android manufacturers. The artefacts of forensic values were identified and analyzed. The result was to help digital forensic investigators and academia in locating and acquiring digital evidence from Kik messenger on android platforms. One of the applications considered in this research; Viber, offers a difficult paradigm of traffic analysis. Hence, Sudozai and Saleem [22] presented a novel methodology for identification of Viber traffic over the network and established a model which can classify its services of audio and audio/video calls, message chats including text and voice chats, group messages and file/media sharing. In an attempt to retrieve correct traffic path, protocols, visible data, certificates and credentials, Network Traffic Forensics on Firefox Mobile OS using facebook, twitter and telegram as case studies were conducted by Yusoff and Dehghantanha [23] using Network Miner 1.0 and Wireshark Portable 1.6.5 to capture the network traffic. The result was the retrieval of Image files, communication texts and authentication credentials. A Comparison of Secure Messaging Protocols and Implementations was conducted by Mujaj [24] where experiments were conducted on real-world implementations of six (6) secure messaging applications to check for Security Properties, Usability and Adoption. All of the applications tested required contact list Upload, five required verification by phone call, phone registration and verification by SMS, four had access to SMS Inbox, could delete devices from account and contained details about transmission of message, three required notification about end-to-end encryption, notification about key changes, QR-code and verified check, two required e-mail Registration, enabled trust-on-first-use, shared keys through 3rd party, passphrase/code, two-step verification and had screen security, one enabled blocking messages, clear trusted contacts and re-encrypt and send message.

Abdul Aziz, Yusof, and AbdRahman [25] use PenDua and Kloner to extract digital evidence from electronic application while FTK and Autopsy with other tools were used for analysis of the extracted evidences. Authors compiled the forensics exercise as a learning package to serve as an eye opener to expose the beginners of Digital Evidence Forensics learners to the tasks. The learning package was tested with 120 students of a Digital Evidence Forensic class for 3 semesters. Majority of the students found the system interesting and best proper procedure of acquiring and analyzing digital evidence. The Saputra, and Riadi [26] research generates information in the form of alerts from attacks displayed by IDS Snort that are already installed on the web server. Authors analyzed the log file using Wireshark for exploration of digital forensics evidence in the form of an IP Address attacks. The results of the analysis using Snort are digital forensics evidence in the form of IP Address and port used by attackers to access the web server. Authors suggested the blocking of IP address and port used by the attacker to access the web server in order to mitigate the attacks. Kumar et al. propose a distributed computing approach for the calculation of network centrality value for each user using the MapReduce approach [27]. Azeez et al. [28] locate phishing sites in order to prevent the users of internet from forms of phishing attacks. It examines the conceptual and literal consistency between the uses uniform resource locator (URL) and the web content.

2.1 Justification of the Study

The justification for this study is to examine the security and privacy of human daily communication using the messaging applications by determining how much information can be reconstructed from the network traffic using traffic capture and analysis tools. The results from this research will be of great importance and significance to developers of both future and existing applications. Measures can be implemented to tighten up the privacy of communication and these measures can be taken note of by future developers. The study can also help in letting users know the extent their communication is secured on different instant messengers; hence they can know what instant messengers should be adopted for confidential communications.

In addition, the tests which were conducted on a wide range of applications in the past were found to have failed at encrypting their data in one way or another. Therefore, there is more need to work in this area. Another reason is the constant changing of these communication applications; the features are added and securities are updated frequently. Facebook Messenger's new in-app downloads are an instance of this. Applications, whether or not they are present in Facebook Messenger, can store data differently depending on user settings, OS version, and manufacturer. Hence, testing needs to be carried out on new versions of these applications/OSs continually as they are released to ascertain what has changed and how much of the prior knowledge of these applications is still valid. Another importance of this research is to encourage both developers and users to care more about security and privacy of their data. It is also needful to mention that new security patches and updates in the communication protocol of any social media application called for re-verification of previous results of the analysis.

3 Methodology

The technique adopts in this research involves data gathering of applications' vulnerabilities from the vulnerabilities analysis and data classification using gathered data to carry out classification of application into malicious or benign. The flow chart in Fig. 1 depicts the process used in carrying out this experiment. In data gathering, the data related to vulnerabilities of application were gathered using the packet capture tools on a Local Area Network, where each of these tools were installed and tested to verify if they are functioning properly as required. If this is true, the tool is used for the packet capture. In addition to using instant messengers to send and receive traffic, each of these messaging applications, was downloaded and installed from the Google Play Store and accounts created or logged into. One of the two smart phones and the host system are connected to the same network while the other smart phone is connected to a different network. Both smart phones then communicate using each instant messenger, while the host system is used for traffic capture and analysis. This analysis reveals the data requires to perform the classification experiment. The second phase was the classification of data gathered from the analysis. This classification was done using the three selected classification algorithms namely Naïve Bayes, Neural network, and Random forest algorithms. The data gathering process and data classification algorithms were discussed in the section below.

The problem is to identify the vulnerability associated with some selected social media communication applications and use these variables to classify the applications into either vulnerable or non-vulnerable. In order to achieve these, selected applications were tested using two selected vulnerability tools namely wireshark and NetworkMiner Capture. The captured vulnerability data were displayed in the Tables 2 and 3. The data were "Yes" or "No" which signify whether an application is susceptible a certain attack or not. The attack could be an application can reveal "text document sent by a sender", "Image", "ISP of sender", "Location of sender", and "Audio". If any of these information can be detected or captured by the vulnerability tools, then the data is "Yes" while "No" is used to represent the scenario where the tools cannot detect the information across the application. After the successful gathering of these data, the data was represented in a binary n by m dimensional vector using "1" for yes and "0" for no data. In this case, 1 was used to represent the presence of vulnerability while 0 was used to represent the absence of vulnerability. The final data was presented for three classification algorithms namely Naïve Bayes, random forest, and neural network. These machine learning algorithms classify the vulnerability data into classification model for identifying vulnerable or non-vulnerable application.

3.1 Experiment Setting

In order to gather data through analysis of application, a TP-LINK wireless router was used to create a Local Area Network (LAN) and to provide internet access for the network. Wireshark, NetworkMiner and Netwitness Investigator were installed on the host system which had a windows10 operating system installed on it. Also, ten (10) instant messaging applications on the Android platform were installed and used for the experiment; the applications were Whatsapp, Viber, Facebook Messenger, Telegram, Imo, Snapchat, BBM, Tango, Skype and Wickr. The applications were randomly chosen with emphasis on popularity and usage. An Ethernet cable was used to connect the router to the internet while the host system was connected to the router's wireless access point. The host system was used in providing mobile hotspot for one smart phone while the other smart phone was connected to the internet through a wireless access point outside the network. Figure 2 displayed the experimental setup for the data gathering process using LAN and combination of tools.

Data Gathering. The data gathered and used were the presence or absence of a particular vulnerability in each of the application. If an application contains a vulnerability or vulnerable to an attack, then "Yes" is used to denote the presence of vulnerability, while "No" is used to denote the absence of vulnerability or an application is not vulnerable to attack. This data was depicted and presented using Table 2 and Table 3. The data was represented using n $\times$ m binary dimensional vector with yes represents presence and no represents absence. The yes and no in the Table 2 and 3 were converted to binary 1 and 0 respectively for effective training and testing of classification model.

3.2 Materials

Data Gathering Materials. The materials used for the data gathering are listed as follows:

I. HOST SYSTEM: HP 2000 running windows 10 operating system, Intel(R) Core (TM) I3-3110M CPU @ 2.40 GHz, 2.40 GHz, installed memory RAM is 6.00 GB.
II. TP-LINK wireless router (Model: TL-MR3420, S/N: 215B439005103, build 150319 Rel.60489n, running firmware version 3.16.9 and hardware version MR3420 V2 00000000, SSID – TP-LINK_4B98)
III. Wireshark-2.6.1 installed on host system
IV. NetworkMiner_2–3-2 installed on host system
V. NetwitnessInvestigator-10.6.1.1.696 installed on host system
VI. A TECNO Camon CX Air (build number - H3713A-N-170618V105, running android version 7.0), along with a gionee m5 mini (build number – [SW VERSION] M5 mini_0301_V8334 [HW VERSION] M5 mini_Mainboard_P3, Android version 6.0)
VII. 9. Instant messaging applications (Whatsapp, Viber, Facebook Messenger, Telegram, Imo, Snapchat, Tango, Skype, Wickr) installed on both smart phones.

3.3 Classification Algorithms

Three classification algorithms used are random forest, neural network, and Naïve Bayes algorithms. These algorithms classify the input data into either malicious or benign application based on the specified classification parameters. The classification forms the model for the new classifier to be used for the identification of application. In the classification process, data were initially converted to numeric binary form and presented in an n by m binary dimensional, where "yes" or "1" represents the presence of vulnerability and "No" or "0" represents the absence of vulnerability. Data was normalized by removing redundant and duplicate. The normalized data were used to feed the classification algorithms to acquire appropriate models. two levels of headings should be numbered. Lower level headings remain unnumbered; they are formatted as run-in headings.

4 Results and Discussion

4.1 Performance Metrics

Statistical tests were used to measure the performance of the proposed model. The metrics like true positive rate, false positive, accuracy, recall, precision and F – measure are used.

TP (True positive) was defined as the vulnerable mobile application that was actually classified as vulnerable i.e. TPR is the proportion of positive instances classified correctly.

TN: non-vulnerable mobile application that was actually classified as non-vulnerable i.e. TNR is the proportion of non-vulnerable instances classified correctly.

FP: Non-vulnerable mobile application that was classified as vulnerable i.e. FPR is the proportion of non-vulnerable instances classified wrongly as positive (vulnerable).

FN: vulnerable mobile application that was classified as non-vulnerable i.e. FNR is the proportion of positive instances wrongly classified as negative (non-vulnerable mobile application.

Therefore:

$TPR = TP/(TP + FN)$ 4.1
$TNR = TN/(TN + FP)$ 4.2
$FPR = FP/(FP + TN)$ 4.3
$FNR = TN/(TN + FN)$ 4.4

The accuracy actually measures the proportion of correctly classified instances (features).

$ACC = (TP + TN)/(TP + TN + FP + FN)$ 4.5

Accuracy.

The accuracy of an algorithm is calculated as the percentage of the dataset correctly classified by the algorithm. It looks at positives or negatives dependently and therefore other measures for performance evaluation apart from the accuracy were used.

$A = (TP + TN)/(TP + TN + FP + FN) * 100\%$ 4.6

Where.

$TP = True\ Positive$
$FP = False\ Positive$
$TN = True\ Negative$

FN = False Negative

Positive and negative represents the classifier's prediction, true and false signify the classifier's expectation.

Precision *Precision = TP/(TP + FP) 4.7*

It indicates the number of instances which are positively classified and are relevant. A high precision shows high relevance in detecting positives.

4.2 Results Presentation

4.3 Result Discussion

The result of the data analysis and classification are presented in Tables 1, 2, and 3. Tables 1 and 2 presented the classification results of the vulnerabilities. This research was able to recover different traffic, including images, documents, and audio. The internet service provider of the sender was also recovered and the resident city of the third party. However, no text/document, image, audio, ISP, Location (Destination City) was recovered from only Viber application. Images were recovered from Facebook messenger, Imo, BBM, Tango and Skype. Texts/documents in different formats like octet-stream, .html, .docx, hexadecimal and plain text were recovered for all applications excluding Viber. The internet service provider of the sender was recovered when testing for WhatsApp, Facebook Messenger, Imo, Snapchat, BBM, Tango and Skype. The location of the receiver was recovered when testing for WhatsApp, Facebook Messenger, Imo, Snapchat, BBM, Tango and Skype. Tango was the only application from which an audio file was recovered. It is also very clear that only Viber messaging application is non-vulnerable while others namely Tango, Facebook Messenger, Imo, BBM, Skype, WhatsApp, Snapchat, Telegram and Wickr are vulnerable to one or more attacks. The Table 1 and 2 shown the classification accuracy, false positive and true positive rates of the models form through classification algorithms earlier mentioned. These results show random forest has better accuracy and false rate prediction than others neural network and naïve bayes, with this research data.

According to the chart in Fig. 3, it can be seen that WhatsApp has 60% vulnerability and 40% security, Facebook Messenger has 80% vulnerability with 20% security, Telegram has 20% vulnerability with 80% security, Imo has 80% vulnerability with 20% security, Snapchat has 60% vulnerability with 40% security, BBM has 80% vulnerability with 20% security, Skype has 80% vulnerability with 20% security and Wickr has 20% vulnerability with 80% security.

Table 1. Classification result of data from NetworkMiner Capture.

Models	TPR	FPR	ACC
Naïve Bayes	0.871921	0.132743	0.938095
Neural Network	0.936585	0.061364	0.938967
Random Forest	0.975432	0.031395	0.969267

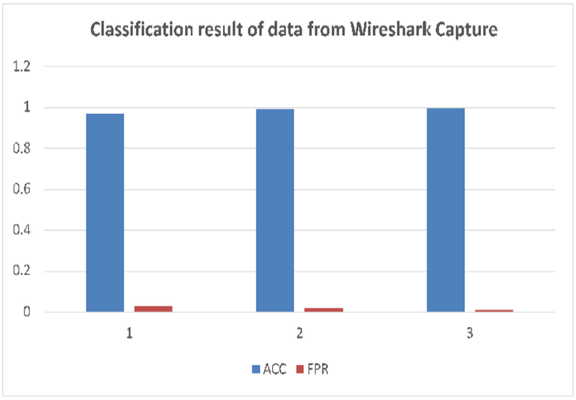

Fig. 1. Classification Accuracy and FPR from Wireshark Capture

Table 2. Classification result of data from NetworkMiner Capture

Models	TPR	FPR	ACC
Naïve Bayes	0.975432	0.031395	0.969267
Neural Network	0.981232	0.020188	0.991125
Random Forest	0.986804	0.014178	0.99381

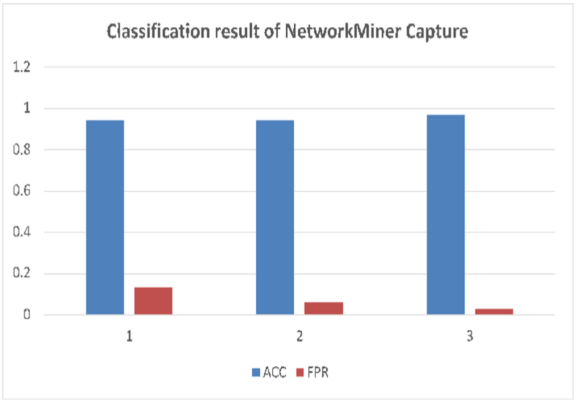

Fig. 2. Classification Accuracy and FPR from NetworkMiner Capture

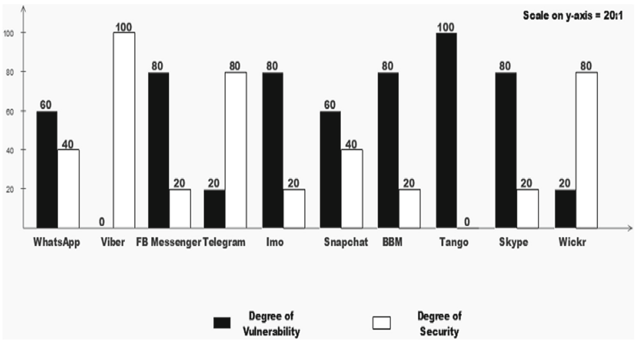

Fig. 3. Vulnerability bar chart

Table 3. Related works Applications vs. Applications used for Experiment

S/N	Applications used in previous Researches	Vulnerability Status before	Applications used in this Research	Vulnerability Status in new Experiment
1	WhatsApp Messenger v.2.11	Non-Vulnerable	WhatsApp – v2.18.267	Vulnerable
2	Audio/Video calls	Non-Vulnerable	Viber – v9.6.5	Non-Vulnerable
3	Facebook	Non-Vulnerable	Facebook Messenger – v3.2.1	Vulnerable
4	Telegram	Non-Vulnerable	Telegram – v1.4.0	Vulnerable
5	Group messages and file/media sharing	Vulnerable	Imo – v9.8.000000010451	Vulnerable
6	Message chats including text and voice chats	Non-Vulnerable	Snapchat – v10.41.5.0	Vulnerable
7	Twitter	Non-Vulnerable	BBM – v3.3.8.73	Vulnerable
8			Tango – v4.9.227627	Most-Vulnerable
9			Skype – Version 8	Vulnerable
10			Wickr Me – v4.55.1	Vulnerable

5 Conclusion

In this research, traffic analyzers were used to investigate and assess the traffic of ten popular messaging applications in order to collect useful data related to vulnerability. Wireshark and Network Miner were used to capture the Network traffic. Both tools were used to ensure the accurate and comprehensive capturing due to their emphasis on different features. Netwitness investigator was used to perform analysis on the captured traffic. The gathered data related to vulnerability were used to build model to classify data into vulnerable and non-vulnerable using machine learning algorithms. The study reveals despite the claim by instant messaging applications about providing end-to-end security, unencrypted data could still be retrieved from the traffic of these applications based on the metrics considered. Viber was shown to be the most secure for private communications while Tango was shown to be the most vulnerable for private communications.

Consequently, sensitive information should be transmitted only through channels that have been proven to be secure and insecure instant messengers should learn from secured ones in other to enhance their security.

6 Recommendation

In view of the overwhelming usage of social media applications, and consider the results of this research which found many of these applications vulnerable it is highly recommended users ensure the adequate privacy setting of the applications. In addition, users must reduce the rate of private or personal data being kept on or sent using these applications. It is also recommended the authors of these applications ensure regular and adequate update of their applications by building patches on the outdated apps.

References

1. Zhou, X., Zhao, Z., Li, R., Zhou, Y., Palicot, J., Zhang, H.: Understanding the Nature of Social Mobile Instant Messaging in Cellular Networks. **18**(3), 389–392 (2014)
2. Andersson, F.: Designing a Secure Client-Server System, September 2009
3. Joshi, M., Hadi, T.H.: A review of network traffic analysis and prediction techniques. *ArXiv Preprint* ArXiv:1507.05722 (2015)
4. Dickson, M., Messenger, M.S.N.: An examination into Trillian basic 3. x contact. Digit. Investig. **4**(1), 36–45 (2007). https://doi.org/10.1016/j.diin.2007.01.003
5. Meghanathan, N., Allam, S.R., Moore, L.A.: Tools and techniques for network forensics. Int. J. Netw. Secur. Appl. (IJNSA), **1**(1). https://arxiv.org/ftp/arxiv/papers/1004/1004.0570.pdf
6. Husain, M.I., Sridhar, R.: iForensics : Forensic Analysis of Instant Messaging on, no. Vim, pp. 9–18 (2010.
7. Chin, E., Felt, A.P., Wagner, D.: Analyzing Inter-Application Communication in Android (2011).
8. Vidas, T., Zhang, C., Christin, N., Vidas, T., Zhang, C., Christin, N.: Towards a general collection methodology for android devices by (2011). https://doi.org/10.1016/j.diin.2011.05.003
9. Zhang, F., He, W., Liu, X., Bridges, P.G.: Inferring Users ' Online Activities Through Traffic Analysis, 59–69 (2011)

10. Appelman, M., Bosma, J., Veerman, G.: Viber communication security. Syst. Netw. Eng. Univ, Amsterdam, Netherlands (2011)
11. Schrittwieser, S., et al.: "Guess Who's Texting You? Evaluating the Security of Smartphone Messaging Applications (2012)
12. Adami, D., Callegari, C., Giordano, S., Pagano, M., Pepe, T.: Skype-hunter: a real-time system for the detection and classification of Skype traffic, no. February 2011, pp. 386–403 (2012). https://doi.org/10.1002/dac.
13. Thakur, N.S.: Forensic Analysis of WhatsApp on Android Smartphones (University Of New Orleans) (2013). https://scholarworks.uno.edu/td/1706
14. Mahajan, A., Dahiya, M., Sanghvi, H.: Forensic Analysis of Instant Messenger Applications on Android Devices. Int. J. Comput. Appl. **68**(8), 38–44 (2013). https://doi.org/10.5120/11602-6965
15. Rafique, M., Khan, M.N.A.: Exploring static and live digital forensics: methods, practices and tools. Int. J. Sci. Eng. Res. **4**(10), 1048–1056 (2013). ISSN 2229–5518
16. Coull, S.E., Dyer, K.P.: Traffic analysis of encrypted messaging services: Apple imessage and beyond. ACM SIGCOMM Comput. Commun. Rev. **44**(5), 5–11 (2014)
17. Anglano, C.: Forensic analysis of WhatsApp messenger on android smartphones arXiv : 1507 . 07739v1 [cs . CR] 28 July 2015, pp. 1–32 (2014). https://doi.org/10.1016/j.diin.2014.04.003.
18. Karpisek, F., Baggili, I., Breitinger, F.: WhatsApp network forensics: decrypting and understanding the WhatsApp call signaling messages, vol. 15, no. October, pp. 110–118 (2015)
19. Abdul Aziz, N., Mokhti, F., Nadhar, M., Nozri, M.: Mobile device forensics: extracting and analysing data from an android-based smartphone. In: 2015 Fourth International Conference on Cyber Security, Cyber Warfare, and Digital Forensic (CyberSec), pp. 123–128. IEEE Publisher (2015).
20. Walnycky, D., Baggili, I., Marrington, A., Moore, J., Breitinger, F.: Network and device forensic analysis of Android social-messaging applications. Digit. Investig. **14**, S77–S84 (2015). https://doi.org/10.1016/j.diin.2015.05.009
21. O. S. Adebayo, S. A. Sulaiman, O. Osho, S. M. Abdulhamid, J. K. Alhassan (2017). Forensics Analysis of KIK Messengers on Android Devices. 2nd International Engineering Conference (IEC: Federal University of Technology. Minna, Nigeria (2017)
22. Sudozai, M.A.K., Saleem, S.: Signatures of Viber Security Traffic Signatures of Viber Secure Traffic, vol. 12, no. 2 (2017)
23. Yusoff, M.N., Dehghantanha, A.: Network Traffic Forensics on Firefox Mobile OS : Facebook , Twitter and Telegram as Case Studies, pp. 63–78 (2017)
24. Mujaj, A.: A comparison of secure messaging protocols and implementations. University of Oslo (2017)
25. Abdul Aziz, N., Yusof, M.S.M., AbdRahman, L.H.: Acquiring and Analysing Digital Evidence - a Teaching and Learning Experience in Class. Cyber Resilience Conference (2018)
26. Riadi, D.S.: Network forensics analysis of man in the middle attack using live forensics method. Int. J. Cyber-Secur. Digit. Forensics (IJCSDF) **8**(1), 66–73 (2019). The Society of Digital Information and Wireless Communications (SDIWC). ISSN: 2305–001
27. Behera, R.K., Rath, S.K., Misra, S., Damaševičius, R., Maskeliūnas, R.: Distributed centrality analysis of social network data using MapReduce. Algorithms **12**(8), 161 (2019)
28. Azeez, N.A., Salaudeen, B.B., Misra, S., Damaševičius, R., Maskeliūnas, R.: Identifying phishing attacks in communication networks using URL consistency features. Int. J. Electron. Secur. Digit. Forensics **12**(2), 200–213 (2020)