ISSN 2708-2032
e-ISSN 2708-2040

INTERNATIONAL UNIVERSITY

INTERNATIONAL JOURNAL OF INFORMATION & COMMUNICATION TECHNOLOGIES

Volume 2, Issue 4
March 2021

ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
MINISTRY OF EDUCATION AND SCIENCE OF THE REPUBLIC OF KAZAKHSTAN

ХАЛЫҚАРАЛЫҚ УНИВЕРСИТЕТІ
МЕЖДУНАРОДНЫЙ УНИВЕРСИТЕТ
INTERNATIONAL UNIVERSITY

INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGIES

МЕЖДУНАРОДНЫЙ ЖУРНАЛ ИНФОРМАЦИОННЫХ И КОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

ХАЛЫҚАРАЛЫҚ АҚПАРАТТЫҚ ЖӘНЕ КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАР ЖУРНАЛЫ

Том 2, Выпуск 8
December 2021

Издание зарегистрировано Министерством информации и общественного развития Республики Казахстан. Свидетельство о постановке на учет No KZ82VPY00020475 от 20.02.2020 г.

Журнал зарегистрирован в Международном центре по регистрации сериальных изданий ISSN (ЮНЕСКО, г. Париж, Франция)

Выходит 4 раза в год.

УЧРЕДИТЕЛЬ:
АО «Международный университет информационных технологий»

ISSN2708-2032 (print)
ISSN2708-2040 (online)

СОДЕРЖАНИЕ

ИНФОКОММУНИКАЦИОННЫЕ СЕТИ И КИБЕРБЕЗОПАСНОСТЬ

Кожахметова Б.А., Губский Д.С., Дайнеко Е.А., Ипалакова М.Т.
Численно-математическое моделирование современных устройств СВЧ и КВЧ диапазонов на примере микрополоскового резонатора.....6

Мубаракова С.Р., Аманжолова С.Т., Ускенбаева Р.К.
Актуальность кибербезопасности в современном мире.....12

Разак А., Әділ А.Ж., Аманжолова С.Т.
Новый инструмент для обнаружения взлома Wi-Fi на основе технологии блокчейн.....18

ЦИФРОВЫЕ ТЕХНОЛОГИИ В ЭКОНОМИКЕ И МЕНЕДЖМЕНТЕ

Аукен В.М.
Анализ взаимодействия государственных доходов и аудита.....38

Бердыкулова Г.М.
Методология преподавания экономических дисциплин в цифровую эру.....42

ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ

Элле В.Ж., Мелисова Л.Т., Куандыков А.А., Куатбаева А.А., Аманбайқызы З.
Свойства реальных бизнес-процессов с точки зрения проектирования.....49

Кошимбай А.Б., Молдагулова А.Н.
Исследование метода анализа и обработки данных социальных сетей с целью определения тональности.....55

Базарбеков И.М., Шарипов Б.Ж.
разработка бизнес-процесса для получения онлайн услуг в организации образования 62

Жунусов Д.О., Алиаскаров С.Ж.
метод классификации текстов на основе алгоритмов машинного обучения.....69

МАТЕМАТИЧЕСКОЕ И КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ

Синчев Б.
О полиномиальной разрешимости класса np-complete.....75

CONTENTS

INFORMATION AND COMMUNICATION NETWORKS, CYBERSECURITY

Kozhakhmetova B.A., Gubsky D.S., Daineko Y.A., Ipalakova M.T.
Numerical and mathematical modeling of modern devices of UHF and EHF bands on the example of a microstrip resonator.....6

Mubarakova S.R., Amanzholova S.T., Uskenbayeva R.K.
Relevance of cybersecurity in the modern world.....12

Razaque A., Adil A. Zh., Amanzholova S.T., Valiyev B.B.
Blockchain technology-featured novel air-cracking tool for wi-fi hacking detection.....18

DIGITAL TECHNOLOGIES IN ECONOMICS AND MANAGEMENT

Auken V.M.
Interaction analysis of government revenue and audit.....38

Berdykulova G.M.
Methodology of teaching the economic disciplines in digital era.....42

INTELLIGENT SYSTEMS

Elle V., Melissova L., KuandykovA.A., Kuatbayeva A.A., Amanbaikyzy Z.
Properties of real business processes from a design point of view.....49

Koshimbay A.B., Moldagulova A.N.
Research method of analyzing and processing social network data in order to determine the tonality.....55

Bazarbekov I.M., Sharipov B.Zh.
development of a business process for obtaining online services in the organization of education 62

Zhunissov D.O., Aliaskarov S.Zh.
method for text classification based on machine learning algorithms 69

MATHEMATICAL AND COMPUTER MODELING

Sinchev B.
On polynomial decision of class NP-complete.....75

МАЗМҰНЫ

АҚПАРАТТЫҚ ЖӘНЕ КОММУНИКАЦИЯЛЫҚ ЖЕЛІЛЕР, КИБЕРҚАУІПСІЗДІК

Кожахметова Б.А., Губский Д.С., Дайнеко Е.А., Ипалакова М.Т.
Микрожолақты резонатор мысалында АЖЖ және ЕЖЖ диапазондарының заманауи құрылғыларын сандық-математикалық үлгілеу.....6

Мубаракова С.Р., Аманжолова С.Т., Ускенбаева Р.К.
Қазіргі әлемдегі кибер қауіпсіздіктің өзектілігі.....12

Разак А., Әділ А.Ж., Аманжолова С.Т.
Блокчейн технологиясына негізделген Wi-Fi хакерін анықтаудың жаңа құралы.....18

ЭКОНОМИКАДАҒЫ ЖӘНЕ МЕНЕДЖМЕНТТЕГІ ЦИФРЛЫҚ ТЕХНОЛОГИЯЛАР

Аукен В.М.
Мемлекеттік кірістер және аудиттің өзара әсерлері.....38

Бердіқұлова Ғ.М.
Цифрлық дәуірде экономиканы оқыту әдістемесі.....42

ИНТЕЛЛЕКТУАЛДЫ ЖҮЙЕЛЕР

Элле В.Ж., Мелисова Л.Т., Куандыков А.А., Куатбаева А.А., Аманбайқызы З.
Жобалау тұрғысынан нақты бизнес-процестердің қасиеттері.....49

Көшімбай А.Б., Молдагулова А.Н.
Тоналдылықты анықтау мақсатында әлеуметтік желілердің деректерін талдау және өңдеу әдісін зерттеу.....55

Базарбеков И.М., Шарипов Б.Ж.
Білім беру ұйымында онлайн қызмет кәрсету үшін бизнес-процессін дамыту.....62

Жунусов Д.О., Алиаскаров С.Ж.
Машинналық оқыту алгоритмдері негізінде мәтіндер классификациясының әдісі.....69

МАТЕМАТИКАЛЫҚ ЖӘНЕ КОМПЬЮТЕРЛІК МОДЕЛЬДЕУ

Синчев Б.
NP-complete сыныптың полиномиялық шешімі туралы.....75

INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGIES
ISSN 2708–2032 (print)
ISSN 2708–2040 (online)
Vol. 2. Is. 4. Number 08 (2021). Pp. 18–37
Journal homepage: https://journal.iitu.edu.kz
https://doi.org/10.54309/IJICT.2021.08.4.003
УДК 004.056.56

Razaque A.*, Adil A. Zh., Amanzholova S.T., Valiyev B.B.
International Information Technology University, Almaty, Kazakhstan

BLOCKCHAIN TECHNOLOGY-FEATURED NOVEL AIR-CRACKING TOOL FOR WI-FI HACKING DETECTION

Abstract. Wi-Fi plays an important role in promoting several application domains such as business, education, industry, etc. On the other hand, if not handled properly, vulnerabilities of Wi-Fi cause damage to the privacy and confidentiality of the users. Some of the hackers use the Linux tool to exploit the vulnerability of Wi-Fi that allows of the hacking process. In this paper, we introduce a Blockchain Technology-Featured Novel Air-Cracking (BTFAT) method to detect the Linux tool for Wi-Fi security improvement. The proposed BTFAT consists of valuable features (e.g., monitoring, scanning, cracking, and testing) which help detect the Linux tool. The BTFAT is programmed on the C platform. Based on the experimental results, the BTFAT produces higher performance as compared to other existing methods.

Keywords: Wi-Fi, vulnerability, BTFAT, privacy, reliability, testing, blockchain technology

Introduction

Wireless networks are now used everywhere. Wi-Fi is used not only by individual users but also by organizations and companies. Wireless networks are embedded in many areas of our life: social networks, business, work, finance (online payments, banking applications) [1-2].

Wi-Fi can make people's daily lives easier, improve the productivity of many companies, and make it easier for employees to work. But there is also a downside, this is the risk of leakage of confidential information through hacking Wi-Fi [6-7]. Many people, users of various social networks and messengers such as Instagram, Facebook, Twitter, WhatsApp, etc., store their data (photos, correspondence, card data) in their accounts. Hackers can hack Wi-Fi through various attacks and use sniffers to intercept traffic, thus gaining access to personal data. The same situation is possible in large business and financial organizations. This can lead to large financial losses for companies or banks [8-9]. Although networks with blockchain technology have a high level of security, they are also susceptible to hacking by intruders. This may lead to the loss of personal data of users or financial losses of companies and organizations [10-11].

As business depends on data, data acquisition speed and accuracy are crucial. The blockchain is perfect for conveying such information because it offers to authorize members of the network an instant, shared and fully transparent access to information in the register unchanged [3]. The blockchain network allows users to track orders, payments, accounts, products, and more. And since all participants share access to a single source of reliable data, it is possible to view all transaction details at any time to work with greater confidence and gain new benefits and opportunities [4-5].

Recently, many studies have been conducted in the field of hacking Wi-Fi using Linux tools, respectively, there are many solutions to this problem. Wi-Fi hacking using the Wireshark traffic analyzer is based on packet capture (PCAP) [12-13], the WPAclean utility uses a four-way handshake method and a beacon to clear capture files [14-15], there are also Linux tools such as Reaver, which uses a WPS connection as a vulnerability to analyze and hack the network [18-19]. The Wifite tool is designed for hacking a network with various encryption algorithms WEP, WPA, WPA2. Wifite uses a set of attacks on Wi-Fi, including brute-force passwords, handshake capture [16-17]. For network hacking, the Wifite tool has flexible settings [20-21]. Motivated by these challenges, the contributions of this paper are summarized as follows:

- the definition of vulnerability for hacking wireless networks using the technology of the Blockchain-Featured Aircrack-ng.
- hacking a wireless network with blockchain technology in practice.

Billions of users and businesses connect to the global network, use Wi-Fi and networks with blockchain technology. As a result, security becomes the most important issue. The main problem is to investigate the vulnerabilities of blockchain networks and based on the detected vulnerabilities, describe recommendations for protection against hacking, so that users and organizations can be less vulnerable to security attacks [22-23].

The following steps should be considered in investigating security issues against Wi-Fi hacking: (a) investigation of various security mechanisms available for WPA/WPA2 using BTFAT, (b) investigation of

vulnerabilities in real-time using the BTFAT, and (c) determining the method of hacking [24-24]. We aim to address these issues and use these solutions in our practical results to make the use of Wi-Fi safer.

The remainder of the paper is organized as follows.

Section II briefly describes the problem and explains its significance. Section III highlights the previous research findings. Section IV describes the state-of-the-art system model. Section V proposes a way of Wi-Fi hacking vulnerabilities using the BTFAT process.

Section VI presents the experimental results and implementation. Section VII gives the discussion of the results. Finally, the conclusions of the paper are presented in Section VIII.

Problem identification

The main problem of this research work is hacking Wi-Fi with Linux using the "Krack" vulnerability (Key Reinstall Attacks). The real problem is researching and finding Wi-Fi vulnerabilities such as incorrectly configured access points, devices with weak encryption keys, impersonating an authorized user. Actions required to resolve this issue:

- to study vulnerabilities and hacking of the Wi-Fi network, then to select the appropriate tool;
- to find the target to attack;
- to check the impact of Pixie dust;
- then to run a full password search. If the PIN code is received but the WPA password is not displayed, to run the commands to get the Wi-Fi password.

Causes for hacking Wi-Fi can be open ports, lack of password protection or weak password protection, lack of data encryption, lack of programs for scanning the network, lack of special services to protect them from attacks. The effects of these causes can be gaining access to the network, interception of network data, commission of various attacks, theft of personal data, interception of passwords, spoofing of the network. The importance of the problem studied in this research paper is that Wi-Fi hacking must be performed as a test of the network and detection of its vulnerabilities to further improve the security of the network perimeter. There are several solutions for hacking Wi-Fi in the form of various attacks such as hacking WPA / WPA2 passwords, attacking WEP, hacking WPS pin, lowering WPA, replacing the true access point with a fake one, fraudulent access point, attacking Wi-Fi access points from global and local networks, denial of service attacks (DoS Wi-Fi), attacks on specific services and functions of routers. An optimistic solution to this problem is to use multiple attacks in combination. This can be implemented using the Linux tool or utility, which includes several or all of the listed kinds of attacks.

Related work

In this section the prominent features of the existing current approaches are summarized.

The main tools for hacking Wi-Fi using Linux are discussed by Sharma [26]. AirSnort uses special algorithms to sort out the password, namely, it analyzes each packet in the network, and when intercepting the required number of data, it decrypts the password from them. AirSnort is available for windows. However, there is one shortcoming - the utility only works with WEP networks.

Bullock & Jeff [27] described the use of packet sniffers Ettercap, Dsniff, and Wireshark for hacking Wi-Fi. Packet sniffers are designed to capture and analyze network traffic. The advantages of traffic analyzers are that they work with the vast majority of known protocols, have a clear and logical graphical interface based on GTK+, and a powerful filter system. Traffic analyzers are also cross-platform and work on such operating systems as Linux, Solaris, FreeBSD, NetBSD, OpenBSD, Mac OS X, and Windows. The disadvantage of these analyzers when hacking Wi-Fi is the need to possess certain skills and abilities in decrypting captured packets. In addition, it is possible to capture packets only in real time.

Li et al. [28] introduces another tool called Reaver for hacking wireless networks that targets certain WPS vulnerabilities. Reaver performs brute force attacks against WPS and registers PIN codes to recover the WPA / WPA2 passphrase. Since many router manufacturers and Internet service providers activate WPS by default, many routers are vulnerable. The disadvantage is that WPS can be disabled.

Wifite is a tool designed to attack multiple wireless networks encrypted using WEP / WPA / WPA2 and WPS. Some parameters are required when WiFite starts working. It records WPA handshakes, automatically disables authentication of connected clients and saves their hacked codes. Hacking Wi-Fi using the Wifite tool is discussed by Sinha [29]. Crunch is a very good and easy-to-use tool for creating custom word lists that can be used in dictionary attacks. Since the success rate of dictionary attacks depends on the quality of the word list, it is impossible to avoid creating your own word lists. The method of hacking the network with the Crunch tool is described by Santo Orcero [30].

MacChanger is a small utility that spoofs a media access control (MAC) address in an arbitrary MAC address. Spoofing the MAC address for Wi-Fi hacking may be necessary to avoid MAC filters or hide the hacker's identity in the wireless network. MacChanger's Wi-Fi hacking approach is discussed by Sinha [31]. After studying these network hacking tools, we have determined that all these tools are essential. However, the above tools have several disadvantages. The disadvantages are that some of these tools can crack only certain encryption algorithms of wireless networks, most of the above tools intercept traffic and hack networks in real time, only at the time of user activity, and have fewer methods for analyzing and hacking a wireless network. But our network hacking tool is not only easy to use, but also has many built-in features for hacking WPA/WPA2/WEP.

System model

The Blockchain technology-featured Aircrack-ng tool is of utmost importance. It successfully detects the Linux tool for Wi-Fi. The BTFAT consists of the features depicted in Figure 1. The features include airdecap-ng, airmon-ng, aireplay-ng, airodump-ng, etc. The airdecap-ng feature decrypts intercepted traffic with a known key, the airmon-ng package puts the network card in the monitoring mode, the airodump-ng feature is a traffic analyzer, it adds traffic to Packet Capture (PCAP) or initialization vectors (IVs) files and shows information about the network. Some of these features are greatly valuable in the Wi-Fi hacking process.

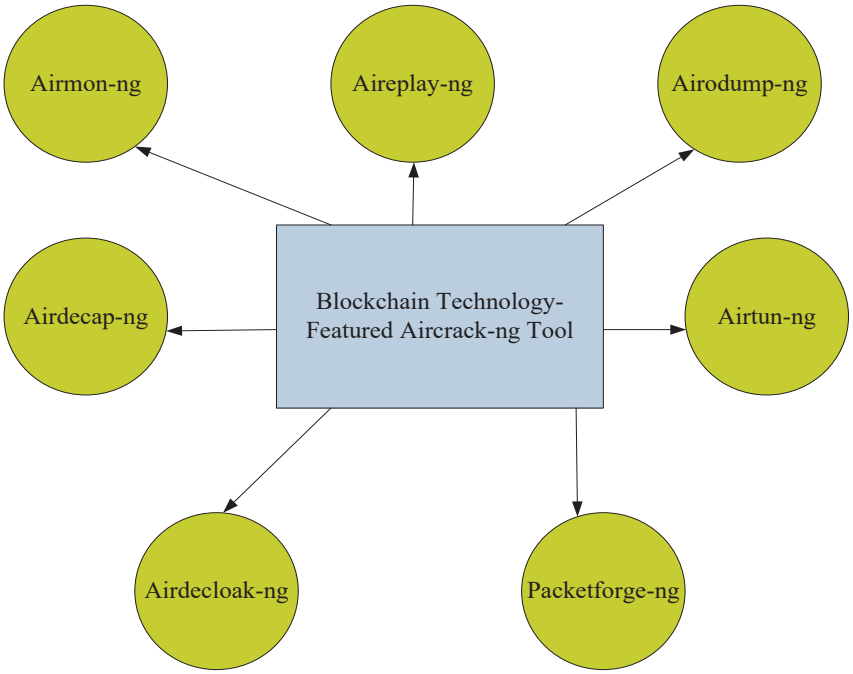

Figure 1 - Components of the Linux BTFAT

To hack Wi-Fi using the BTFAT, the hacker first connects to the Wi-Fi adapter and determines the network interfaces. To do this, the airmon-ng package defines the available network interfaces, as well as the driver. If the network interface driver is detected as a result of the command execution, the network is monitored. Otherwise, the driver is debugged. Network monitoring is performed by the airmon-ng feature as a result of network monitoring, a message should appear indicating that the monitoring mode was successfully enabled on the previously defined interface. Then, using the airodump-ng feature, the listening mode is enabled to determine the available Wi-Fi networks. As a result, the screen displays a list of wireless networks within the range of the Wi-Fi adapter. The screen also displays important characteristics for network hacking, such as the encryption used (WEP, WPA/WPA2), channel, and basic service set id (BSSID). Knowing the necessary information about the network, packets are captured using the airodump-ng package containing the encrypted password. When capturing packets, it is important to capture many IVs packets over 1000. The waiting time depends on the network activity. If no one is connected to the access point, the time may be delayed.

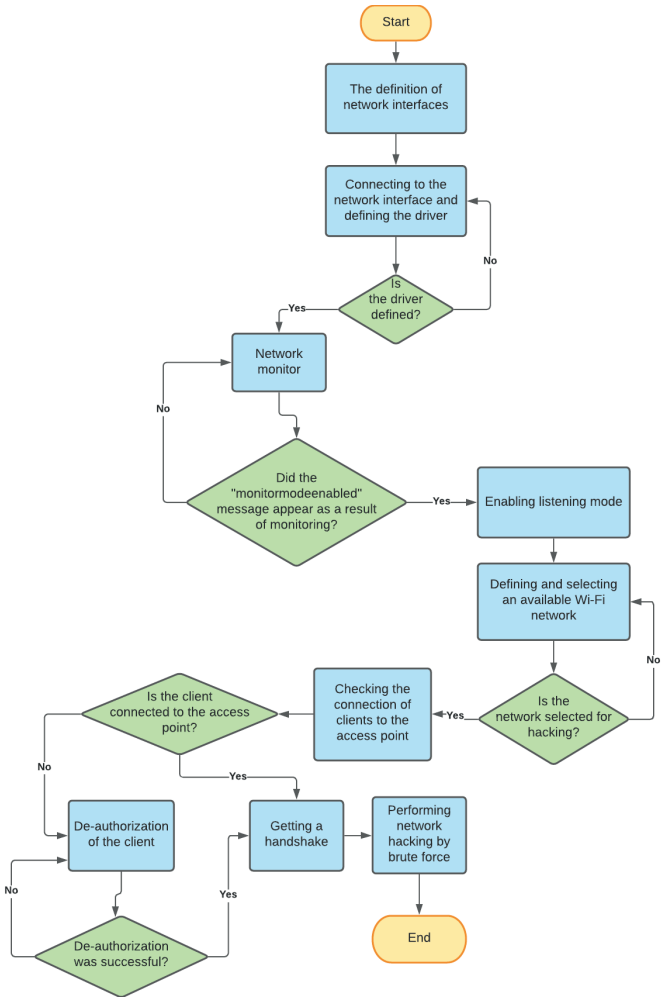

Figure 2 - The process of hacking Wi-Fi using the BTFAT

The distance to the access point is not as important as the network activity. To reduce the time for collecting packets, the client logs in. After successful de-authorization of the client, the hacker receives an intercepted handshake. Next, the hacker performs a brute-force hacking using a password dictionary. The process of hacking Wi-Fi using the aircrack-ng tool is depicted in Figure 2.

Figure 3 explains the system model of the Wi-Fi hacking process using the BTFAT and depicts a second (fake) access point created by the hacker during Wi-Fi hacking process. Using this access point, the hacker de-authorizes the user through multiple requests. After reconnecting the user from the real network to the access point created by the hacker, the hacker initiates the handshake. Based on the received handshake, it is possible to hack the network and decrypt the password.

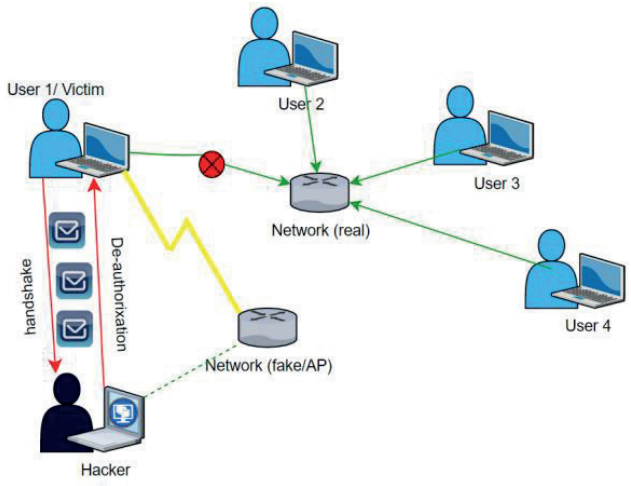

Figure 3 - System model

Proposed Wi-Fi hacking using the BTFAT process

The method proposed for hacking Wi-Fi uses the BTFAT. Before the Wi-Fi is hacked, methods are studied to protect the network. To protect Wi-Fi networks, several well-known methods are used, such as access restriction and authentication methods. This research paper discusses the method of hacking Wi-Fi, which uses the authentication method as a network protection. In turn, authentication methods for network protection are classified: open authentication, Shared Key Authentication (WEP encryption), Mac address authentication, Wi-Fi protected access (WPA), Wisconsin-Internet protected Access2 (WPA2), Cisco Centralized Key Management (CCKM). The BTFAT breaks WEP, WPA, and WPA2 keys. The process of hacking Wi-Fi with the BTFAT consists of three phases:

- packet-capturing and saving processes
- client de-authorization process
- Wi-Fi blockchain-featured hacking process

A. *Packet-capturing and saving processes*

This process is implemented at the beginning and is necessary for collecting IVs data packets. During this process, the network is monitored, as a result of which there are available network interfaces. After that, the hacker connects to them and captures the packets. Then all packages are saved in a single file. IVs packets contain the necessary information to decrypt the password of the required network. Packet-capturing and saving processes are presented in Table 1.

Table 1 - Packet-capturing and saving processes

Algorithm-1: Packet-Capturing and Saving Processes

1. Initialization: {N_c: Network Channel; M_{pa}: MAC address of Access Point; I: Interface; P_{cf}: Packets-captured file; L_t : Linux tool; N_m: Network monitoring; N: Network; F_0: Folder; P: Packets}
2. Input: {N_c, M_{pa}, I}
3. Output: {P_{cf}}
4. Set N_c, M_{pa}, I
5. Do Process $N_m \in N \leftarrow L_t$
6. While $N_m \in N<1$
7. Capture P
8. Sum $P_{cf} = P + 1$
9. Do $N_m = 0$
10. Save P_{cf} to F_0
11. End while

Algorithm-1 explains the packing capturing and saving processes. In step 1, variables are initialized for packet capturing and saving. Steps 2-3 explain the input and output variables respectively. Step 4 uses the components (e.g., network channel, physical address of the access point and interface) for network monitoring process. Step 5 shows the process of using Linux tool on the network for network monitoring process. Steps 6-9 shows the entire network monitoring process and attempts to capture the packets, which are stored into the packet-capturing list. This process continues until the entire network is monitored and all of the packets are stored into the packet-capturing list. In step 10, the packet-capturing list is saved into folder for further process.

There are several properties that define packet capture:

- the total time it takes to capture packets;
- the average interval between adjacent packets;
- the average packet waiting time.

Definition-1: the average value of the interval between adjacent packets τ_a is the average time of packet captures between the previous and subsequent packets and is calculated by the equation (1):

$$\tau_a = \frac{1}{M} \times \sum_{s=0}^{M} (a_{t+1} - a_t) \qquad (1)$$

Where α_t: moments of time when packets arrive; M: number of analyzed intervals.

Theorem-1: The higher the load on the connection channel, the longer is the total time required to capture packets.

Proof: The channel load factor is calculated by the equation (2):

$$L_c = \frac{\sum P_t}{\sum E_p} \quad (2)$$

Where P_t: capture time of the packet; E_p: end time of processing of the *i*-th packet.

The number of packets and their size (in bytes) and the time of traffic measurement are known. Then, the total capture time of the packet is equal to:

$$\sum P_t = \frac{(B+N)\times 8}{V} \quad (3)$$

Where P_t: capture time of the packet; *B*: number of bytes transmitted; *N:* number of packets captured; *V:* packet capture rate.

The total processing time of the *i*-th packet is equivalent to the time of traffic measurement and is determined by the equation (4):

$$\sum E_p = \varphi \quad (4)$$

Where φ: the time of traffic measurement.

Based on the previous equations, the channel load factor is equal to:

$$L_c = \frac{(B+N)\times 8}{V\varphi} \quad (5)$$

Where *B:* number of bytes transmitted; *N:* number of packets captured; *V:* packet capture rate; φ: the time of traffic measurement.

Thus, the higher the channel load factor, the longer the packet capture time.

Hypothesis-1: The higher the packet intensity detected during network monitoring, the shorter is the packet capture time.

Proof: The packet capture time can be determined by the equation (6):

$$T_c = \frac{I_p}{1-M_a\times I_p} \quad (6)$$

Where I_p: packet intensity (packets/sec); M_a: average network monitoring time.

Let the packet capture time be expressed in terms of traffic intensity T_I and packet length L_p, and channel throughput T_h:

$$I_p = \frac{T_I}{L_P} \quad (7)$$

Where T_I: traffic intensity; L_p: packet length.

The average network monitoring time is determined by the equation (8):

$$M_a = \frac{L_P}{T_C} \quad (8)$$

Where L_p: the packet length; T_h: the channel throughput; M_a: the average network monitoring time.

Then, the equations (7) and (8) are substituted in the equation (6):

$$T_h = T_I + \frac{L_P}{T_c} \quad (9)$$

Where T_h: the channel throughput; T_I: the traffic intensity; L_p: the packet length; T_c: the packet capture time.

Based on the above equations, corollary-1 is derived.

Corollary-1: To reduce packet capture time, the bandwidth of the channel must be high.

B. *Client de-authorization process*

After finding the network interfaces and selecting an access point for hacking, a handshake should be conducted. To receive a handshake, the user must be active on the network. If there is no activity, the activity is created by deactivating the client. During the client deactivation process, the access point (fake) sends requests to the client until the client reconnects to the network. Thus, if deactivation is successful, the hacker receives a handshake. Client de-authorization and handshake recording process are given in Table 2.

Table 2 - Client de-authorization and handshake recording process

Algorithm-2: Client de-authorization and handshake recording process

1. Initialization: {A_{pc}: client's physical address; A_{pa}: physical address of the access point; H: handshake; I: interface; C: client; A_p access point; S: client's SSID; P: password; R_s: recconnect, P_{cf}: packets-captured file}

2. Input: { A_{pc}, A_{pa}, I }

3. Output: {H}

4. Set A_{pc}, A_{pa}, I

5. A_p requests $\Rightarrow C \rightarrow R_s$

6. **While** $A_p = R_s$

7. **Do** $A_p \leftarrow H \in (P, S)$

8. **Record** H to P_{cf}

9. **End while**

Algorithm-2 explains the client de-authorization and handshake recording processes. In step 1, the variables are initialized for the process of client de-authorization and handshake recording. Steps 2-3 give the input and output processes, respectively. Step 4 uses the network components (e.g., client's physical address, physical address of the access point, interface) for implementing requests. In step 5 requests are sent from the access point to the client to reconnect to the network. Steps 6-7 explain passing the handshake, which includes the password and client ID number to the access point. This process continues while the client is reconnecting to the network. In step 8 the received handshake is written into the captured packets that were received during network monitoring in the previous algorithm for further use in the Wi-Fi hacking process.

The time of de-authorization is characterized by the following properties:

- the total time of sending requests to the user;
- the total intensity of answers received by the hacker;
- processing of responses received from the user and establishing a handshake.

Definition-2: The total intensity of responses received by the hacker β_T is the sum of the intensity of the flow of requests sent to the user $\beta_H = (1 - C) \times \beta$ and the intensity of processed responses sent by the user $\beta_U = (1 - C) \times \beta$ and is calculated by the equation (10):

$$\beta_T = \beta + (1 - C) \times \beta \qquad (10)$$

Where β: the intensity of the elementary stream requests; C: probability of self-classification of a new request stream by a second access point.

Theorem-2: The intensity of sending requests by the hacker affects the performance of processing requests by the user and the average delay in sending requests.

Proof: The performance of processing requests by the user (P_R) is determined by the equation (11):

$$P_R = \frac{\beta + (1 - C) \times \beta}{\omega} \qquad (11)$$

Where P_R the performance of processing requests by the user ω: the intensity of the query processing; β: intensity of sending requests; C: probability of self-classification of a new request stream by a second access point.

The probability that the communication channel for sending the request is free (P_c) can be obtained by the equation (12):

$$P_C = \frac{1}{\frac{P_R^{m+1}}{m! \times (m - P_R)} + \sum_{m=0}^{m} \frac{P_R^m}{m!}} \qquad (12)$$

Where m: the number of processors.

The average delay in sending requests (D_A) can be obtained based on the number of requests sent (S_R), depending on the average number of requests in the queue (Q_A):

$$Q_A = \frac{P_R^{m+1} * P_C}{mm! \, (1 - \frac{P_R}{m})^2}, \qquad (13)$$

$$S_R = Q_A + P_C \ , \qquad (14)$$

$$D_A = \frac{S_R}{\beta + (1 - C) \times \beta} \qquad (15)$$

Where *m:* the number of processors; P_c*:* the probability that the communication channel for sending the request is free; P_R*:* the performance of request processing by the user; β*:* the intensity of the elementary stream requests; *C:* the probability of self-classification of a new request stream by a second access point.

Hypothesis-2: The smaller the volume of transmitted requests, the longer it takes for a hacker to get a handshake.

Proof: Each request has the same length and requires a transmission time (τ_T). The time of transmission of the message about the client's acceptance of the request is assumed to be equal to τ_R. The time for sending a request (τ_S) is calculated using the equation (16):

$$\tau_S = N \times \tau_T + \tau_R + N \times \tau_A + \tau_W \quad (16)$$

Where τ_T: the transmission time of the request; N: the number of requests; τ_A: the average processing time of the response received by the hacker; τ_W: the average waiting time for a request in the queue until the communication line is free; τ_R: the time of transmission of the message about the client's acceptance of the request.

Since the bandwidth of the communication channel and the average length of each request are known, the average time for its transmission can be determined by the equation (17):

$$\tau_T = \frac{R_v}{C_h} \quad (17)$$

Where R_v*:* a known volume of the request in bits; C_h*:* channel capacity bit/sec.

The time of transmission of the message about the client's acceptance of the request is calculated similarly by the equation (18):

$$\tau_R = \frac{R_A}{C_h} \quad (18)$$

Where R_A*:* known volume of the request acceptance message; C_h*:* channel capacity bit/sec.

To calculate the average waiting time τ_W and the average message processing time τ_A, it is assumed that the input stream of packets from the user forms a simple stream with an average intensity μ, and the average service time A_S calculated by the equation: (19):

$$A_s = \frac{1}{\gamma} \quad (19)$$

The request received in the buffer will wait until the communication line is released, i.e. until the processing of the message about the acceptance of the previous request is completed. Probabilities of finding a packet in a buffer queue of infinite length is calculated by the equation (20):

$$Q(N,L) = \frac{\frac{L^N}{N!} * \frac{1}{1 - L/N}}{\sum_{k=0}^{N-1} \frac{L^k}{k!} + \frac{L^N}{N!} * \frac{N}{N-L}} \quad (20)$$

Where $L = \frac{\mu}{\gamma}$: full input load.

The average number of requests can be found by the equation (21):

$$A_N = \frac{L}{N-L} * Q(N,L) \quad (21)$$

Where A_N: the average number of requests; Q*(N,L)*: the probabilities of finding a packet in a buffer queue of infinite length; *L*: the full input load.

Based on the previous equations, the average waiting time for a request in the queue is calculated by the equation (22):

$$\tau_W = \frac{A_N}{\mu} = \frac{Q(N,L)}{\gamma(N-L)} \quad (22)$$

Where A_N: average number of requests; μ: average intensity.

The average processing time of a single request is determined by the equation (23):

$$\tau_A = \frac{L}{\mu} = \frac{1}{\gamma} \quad (23)$$

Where *L:* full input load; μ average intensity.

Thus, if the parameters τ_W and τ_A are unchanged, the time of sending the request τ_S is determined by the equation (24):

$$\tau_S = \frac{N * R_v}{C_h} + \frac{R_A}{C_h} + \frac{N}{\gamma} + \frac{Q(N,L)}{\gamma(N-L)} \quad (24)$$

Where R_v: the known volume of the request in bits; C_h: the channel capacity bit/sec; R_A: the known volume of the request acceptance message; Q*(N,L):* the probabilities of finding a packet in a buffer queue of infinite length; *L:* the full input load, N: the number of requests.

Corollary-2: To reduce the volume of transmitted requests, it is necessary to transmit requests of greater length to speed up the time of receiving the handshake. This corollary was based on the analysis of equation (24).

Considering that all requests have equal length and average transmission time, the duration of the communication channel occupation when transmitting one request after establishing a connection between the hacker and the user is determined by the equation (25):

$$\tau_H = N \times \tau_T + N \times \tau_A + \tau_M \quad (25)$$

Where $R_M = \tau_M \times C_h$: the volume of transmitted requests.

Thus, the total time for sending requests is determined by the equation (26):

$$\tau = \tau_S + \tau_H = N \times \tau_S + N \times \tau_H + \tau_M + \tau_R + \tau_W \quad (26):$$
$$= \frac{N * R_v}{C_h} + \frac{N}{\gamma} + \frac{R_A}{C_h} + \frac{R_M}{C_h} + \frac{Q(N,L)}{\gamma(N-L)}$$

Where τ_S: the time for sending a request; R_v: the known volume of the request in bits; C_h: the channel capacity bit/sec; R_A: the known volume of the request acceptance message; Q*(N,L):* the probabilities of finding a packet in a buffer queue of infinite length; *L:* the full input load, N: the number of requests.

C. *Wi-Fi Blockchain-Featured Hacking Process*

The last phase is hacking the wireless network using BTFAT. Blocks in the blockchain system can only create a certain number of bitcoins, transactions must have a certain format and correct signatures for spent bitcoins, a transaction cannot be performed twice within the same blockchain, etc. The blockchain cannot be hacked by attacking the encrypted traffic of an individual node: if the consensus rules are violated in the block, the blockchain system denies the operation of an individual node, even if other nodes believe that an intrusion into the chain of records did not occur.

To hack network with blockchain technology, ARP spoofing is performed after capturing the network traffic. This is an attack committed when sending ARP messages to the local network. The purpose of this attack is to link the hacker's MAC address to the IP address of another host, such as the default gateway. Thus, any traffic directed to a specific IP address is sent to the hacker. After making an attack on the network, the hacker inserts a malicious script into the HTML pages that the user views the command “to call the miner” and deploys an HTTP server on its computer to serve the miner. Figure 2.1 depicts the process of hacking Wi-Fi with BTFAT. The goal of the third phase is to carry out an autonomous attack on the Wi-fi network to introduce a malicious code. With the help of the built-in BATTAT tools, it is possible to analyze and edit traffic. For the purity of the hacking process, only one line of code is embedded in the HTML page, which calls the miner. After the traffic is captured, the JavaScript code is embedded in it, and an injector is created. The created injector adds a string to the HTML with a call to the JavaScript miner. The packet-capturing and saving processes are shown in Table 3.

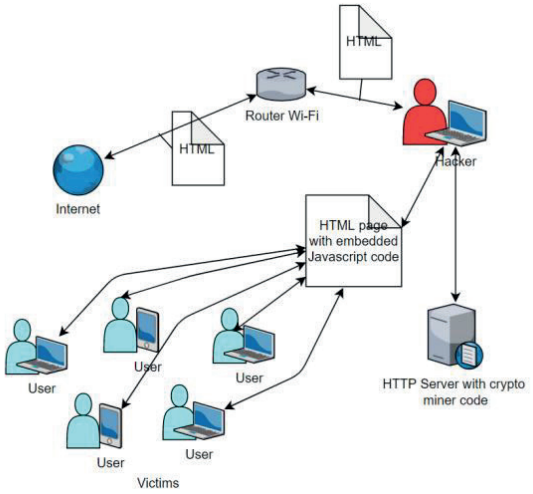

Figure 2.1 - The process of hacking Wi-Fi with BTFAT

Table 3 - Wi-Fi Hacking process with BTFAT

Algorithm-3: Wi-Fi hacking process with BTFAT

1. Initialization: {T_N: captured network traffic; U_{ip}: user's IP address; H_a: hacker's MAC address, N: wireless network, E_m: embedded in the network miner; A_{sp}: ARP spoofing; P_{ht}: HTML pages, S: iterate through all pages, JS_c: JavaScript code; M_s: getting started miner; H_c: hacker's computer}

2. Input: { T_N, U_{ip}, H_a, N}

3. Output: {E_m}

4. Set U_{ip}, H_a, N

5. Do process $A_{sp} \rightarrow N$

6. Link U_{ip} to H_a

7. For $P_{ht}=0\ to\ P_{ht} = S$

8. If $P_{ht} \in T_N$

9. Set $JS_c \rightarrow T_N = M_s$

10. End if

11. Deploy $P_{ht}\ on\ H_c$

12. Do $JS_c \rightarrow P_{ht}$

Algorithm-3 explains the process of hacking Wi-Fi with BTFAT. In step 1, the variables are initialized for the process of hacking Wi-Fi. Steps 2-3 give the input and output processes, respectively. In Step 4, the network for hacking, the user's IP address and the hacker's MAC address are determined. Step 5 is an ARP spoofing attack on the network. Step 6 is linking the hacker's MAC address to the user's IP address to direct traffic from the user's IP address to the hacker. Step 7 is iteration through each page directed to the user. In Step 8 we check that the HTML page belongs to the captured traffic. In Steps 9-10 we embed the JavaScript code into the captured traffic, thus triggering the miner. In Step 11, we deploy the HTTP server on the hacker's computer to serve the miner. In Step 12, JavaScript code is embedded in HTML pages for mining.

A large amount of data contained in the captured packets is analyzed in order to get the password.

Condition: m packets are used to analyze n amounts of data. Let's make the parameters of i-th packets as $b_i (i = 1,2,3, \dots, m)$. This parameter is set to analyze packets by parameters when searching for an encrypted password.

The analysis is performed for each packet from the 1st to the i-th, and a single volume of the j-th amount of data is used. Let the use of the i-th order packet to analyze a single volume of the j-th amount of data be given as $a_{ij} (i = 1,2,3, \dots, m; j=1,2,3, \dots, n)$.

Packet analysis takes a certain time the amount of which depends on the speed of analysis. Therefore, let the speed from analyzing the unit volume of the j-th amount of data be set as $c_j (j = 1,2,3, \dots, n)$.

This mathematical model explains a data analysis plan that provides the maximum speed of analysis under the specified restrictions on data packets.

The volume of analysis of the j-th amount of data is set as a vector of variables equal to $X = (X1, X2, X3, \dots, Xn)$, where $x_j (j = 1,2,3, \dots, n)$ is the volume of analysis of the j-th type of data.

When analyzing packets, a restriction is imposed on their number. It follows from the variable vector that the restriction on using packages for analyzing all data is set by the expression:

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i \qquad (27)$$

Based on the speed of analysis of the j-th data quantity $c_j x_j$, the objective function is calculated using the equation (28):

$$Z(X) = c_1x_1 + c_2x_2 + \cdots + c_nx_n \qquad (28)$$

A mathematical model for analyzing the m number of packets that contain n amount of data is defined as a system of the following expressions:

$$M = \begin{cases} Z(X) = c_1x_1 + c_2x_2 + \cdots + c_nx_n \rightarrow max, \\ a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leq b_1, \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \leq b_2, \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots. \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \leq b_m, \\ x_j \geq 0, j = 1,2,3, \dots, n. \end{cases} \qquad (29)$$

Where M: the process of analyzing packages; m: the number of the packets; n: the amount of data.

Let the set of attacks made by the BTFAT be given by the expression (30):

$$K \in K_1 \times K_2 \times K_3 \ldots \times K_b \quad (30)$$

Where $K_b\left(\overline{1, b+1}\right)$: the set of values of the i-th parameter of a particular attack that determines the type of attack. Each attack $\vec{k} \in K$ is a vector $(k_1, k_2, \ldots, k_{b+1})$, where $\overrightarrow{k_b} \in K_b$.

Rainbow tables are defined as an expression (31):

$$\vec{y} \in Y, Y \in Y_1 \times Y_2 \times Y_3 \ldots \times Y_j \quad (31)$$

Where $Y_j(j = \overline{1,n})$: set of values of the j-th parameter of the rainbow table.

The network for hacking is indicated by the expression (32):

$$\vec{g} \in G, G \in G_1 \times G_2 \times G_3 \ldots \times G_f \quad (32)$$

Where $G_f(f = \overline{1,m})$: the set of values of the f-th parameter of the wireless network.

The success of hacking the network using the BTFAT is related to the attack used to break into the wireless network and the formation of rainbow tables in the process of decrypting the password. Thus, the function that sets the level of successful hacking of the network by an attack $\vec{k} \in K$ c the application of rainbow tables $\vec{y} \in Y$ to hack the wireless network $\vec{g} \in G$ is denoted by the expression (33):

$$\delta: K \times Y \times G \to [0; 1] \quad (33)$$

Where δ: the function that sets the level of successful hacking; K: the attack; Y: the rainbow tables; G: the wireless network.

The function that determines the degree of success from applying an attack to a wireless network is calculated by the expression (34):

$$\beta: G \times K \to [0; 1] \quad (34)$$

Where β: the function that determines the degree of success from applying an attack to a wireless network; G: the wireless network; K: the attack.

The probability of a successful application of a hacker attack with rainbow tables is calculated:

$$\gamma: Y \times K \to [0; 1] \quad (35)$$

Where γ: the probability of a successful application of a hacker attack with rainbow tables; K: the attack; Y: the rainbow tables.

Thus, based on the expressions (33), (34), (35), the function δ is expressed as:

$$\delta\left(\bar{k}, \bar{y}, \bar{g}\right) = \beta\left(\bar{g}, \bar{k}\right) * \gamma\left(\bar{y}, \bar{k}\right) \quad (36)$$

Where $\delta\left(\bar{k}, \bar{y}, \bar{g}\right)$: the function that sets the level of successful hacking; $\beta\left(\bar{g}, \bar{k}\right)$: the function that determines the degree of success from applying an attack to a wireless network; $\gamma\left(\bar{y}, \bar{k}\right)$: the probability of a successful application of a hacker attack with rainbow tables.

Define the function $\beta\left(\bar{g}, \bar{k}\right)$. To do this, consider a family of functions:

$$\beta_{uh}: G_g \times K_h \to R_+ \quad (37)$$

Where R_+: the set of non-negative real numbers; β_{uh}: a function that sets the level of mutual influence of the wireless network parameter G_g and the attack parameter k_h on the network:

$$\beta_{uh}(g, k) = 0, \quad (38)$$

if an attack with the value of the parameter $k \in K_h$ is not applicable to a wireless network with the $c \in G_g$ parameter value.

$$0 < \beta_{uh}(g, k) < 1, \quad (39)$$

if the value of the wireless network parameter $c \in G_g$ reduces the probability of a successful attack with the value of the parameter $k \in K_h$.

$$\beta_{uh}(g, k) = 1, \quad (40)$$

if the value of the wireless network parameter $c \in G_g$ does not affect the applicability of the attack with the parameter $k \in K_h$.

$$\beta_{uh}(g, k) > 1, \quad (41)$$

if the value of the wireless network parameter $c \in G_g$ indicates that an attack with the parameter $k \in K_h$ is applicable for hacking.

Denote by $\overline{\beta_{uh}}: G_g \times K_h \to [0;1]$ the function:

$$\overline{\beta_{uh}}(g,k) = \frac{\beta_{uh}(g,k)}{\sum_{\varepsilon \in C_g} \beta_{uh}(\varepsilon,k)} \quad (42)$$

Then, based on the expression (18), the success rate of applying the attack $\vec{k} \in K$ to the wireless network $\vec{g} \in G$ is calculated:

$$\beta(\vec{g},\vec{k}) = \min_{h=1,b+1} \prod_{g=1,s} \overline{\beta_{uh}}(g_u,k_h) \quad (43)$$

Where the attack and wireless network are set by the parameters $(k_1, k_2, \dots, k_{b+1})$ and $(g_1, g_2, \dots, g_f)$, respectively.

The function $\gamma(\bar{y},\bar{k})$ is expressed similarly to the function $\beta(\vec{g},\vec{k})$:

$$\gamma(\bar{y},\bar{k}) = \min_{h=1,b+1} \prod_{t=1,s} \overline{\gamma_{th}}(y_t,k_h) \quad (44)$$

Where the attack and rainbow table are set by the parameters $(k_1, k_2, \dots, k_{b+1})$ and $(y_1, y_2, \dots, y_j)$, respectively.

Thus, the function that sets the level of successful hacking of the network by an attack $\vec{k} \in K$ c the application of rainbow tables $\vec{y} \in Y$ to hack the wireless network $\vec{g} \in G$ takes the form:

$$\delta(\bar{k},\bar{y},\bar{g}) = \min_{h=1,b+1} \prod_{g=1,s} \overline{\beta_{uh}}(g_u,k_h) * \min_{h=1,b+1} \prod_{t=1,s} \overline{\gamma_{th}}(y_t,k_h) \quad (21)$$

The reliability of Wi-Fi hacking is characterized by the probability of password decryption, which is determined by the equation (45):

$$P_c(t) = \frac{N_0 - \sum n_i}{N_0} \quad (45)$$

Where $P_c(t)$: reliability of Wi-Fi hacking; N_0: the number of initially captured packets; $\sum n_i$: the number of denied de-authorization requests.

The probability of decrypting the password from the received handshake is equal to the product of the probabilities of successful processing of elements of the Wi-Fi hacking process (packet capture, requests for client deauthorization, half-baked handshake):

$$P_c = P_1 \times P_2 \times P_3 \dots \times P_n \quad (46)$$

Where P_c: the probability of decrypting the password; P_n: the probabilities of successful processing of elements of the Wi-Fi hacking process.

Theorem-3: The time of cracking the Wi-Fi (T_c) depends on the complexity of the password, which is selected from the space of possible passwords ($P = L^c$).

Proof: A password is selected from the space of possible passwords. The size of the space P is determined by the expression (47):

$$P = L^c \quad (47)$$

Where P: the size of the possible password space; L: the length of characters in the password; C: the number of characters in the password.

Thus, the time is calculated by the expression (48):

$$T_c = \frac{P}{10^9 \times 3600} \quad (48)$$

Where T_c: the time of cracking Wi-Fi; P: the size of the possible password space.

Hypothesis-3: Hacking a network using the BTFAT tool takes less memory, less processing power, and less time as compared to other tools designed to hack a network.

Proof: To break into the network, a hacker needs to get a handshake containing an encrypted password and to decrypt the password. The W function converts the encrypted password $e(P)$ into a new password $W(e(P))$. The encrypted password in the handshake is written in binary notation, and the password is written as numbers in the notation Q, where Q: the number

of possible characters for passwords. The C function then converts the data from the binary number system to the Q number system. For each encrypted password $e(P)$, the function calculates a new password $W(e(P))$. The BTFAT for hacking a wireless network has the ability to use rainbow tables, which speed up the process of decrypting the password while spending less computer resources. Using the W function, it is possible to precompute data tables (rainbow tables).

To generate a data point in the rainbow table, a possible password P_0 is assigned, an encrypted password $e(P_0)$ is calculated, then a possible password $W(e(P_0))$ is calculated, which becomes P_1. This process continues until the encrypted password starts with twenty 0 ($e(P_n)$). Such an encrypted password occurs 1 time in about 10^6 encrypted passwords. The pair $[P_0, e(P_n)]$ that contains an encrypted password starting with twenty 0 is stored in the table.

The set of such pairs is calculated. Each pair contains a sequence of possible passwords $P_0, P_1, \dots, P_n$ and encrypted passwords. However, there may be spaces, meaning some passwords may not be present in all calculations. For a good database without spaces, the memory required to store the calculated pairs is small. Presumably, in the captured packets, passwords have a certain type: 12 characters, taken from 26 letters of the alphabet. The encrypted password d_0 in the captured packet data set is used to identify the associated password. To do this, first calculate $e(W(d_0))$ to get the new encrypted password d_1, then calculate $e(W(d_1))$ to get d_2, and so on until the encrypted password starting with twenty 0 (d_m) is displayed. The table is then checked to see which source password, P_0, the encrypted fm password is associated with. Based on P_0, the password and encrypted passwords $e_1, e_2, \dots$ are calculated until the original encrypted password d_0, denoted d_k, is generated. The password that the hacker is looking for is the one that gave rise to d_k, i.e. $(W(d_k - 1)$, which is one step earlier in the chain of calculations.

The required computation time is what it takes to find the d_m in the table plus the time it takes to compute the sequence of encrypted passwords from the corresponding password $(e_1, e_2, \dots, e_k)$ which is about a million times less than the time it takes to compute the table itself. Thus, performing a preliminary calculation and storing the results allows a hacker to get any password with a known encrypted password in a reasonable amount of time. This process takes a few seconds.

The process of decrypting the password by the rainbow tables of the novel BTFAT is presented in expression (26):

$$R_t = \begin{cases} G_0 \overset{e}{\rightarrow} e(G_0) \overset{W}{\rightarrow} G_1 \overset{e}{\rightarrow} e(G_1) \overset{W}{\rightarrow} G_2 \overset{e}{\rightarrow} \dots \overset{e}{\rightarrow} e(G_n) \\ F_0 \overset{e}{\rightarrow} e(F_0) \overset{W}{\rightarrow} G_1 \overset{e}{\rightarrow} e(F_1) \overset{W}{\rightarrow} F_2 \overset{e}{\rightarrow} \dots \overset{e}{\rightarrow} e(F_n) \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots. \\ J_0 \overset{e}{\rightarrow} e(J_0) \overset{W}{\rightarrow} G_1 \overset{e}{\rightarrow} e(J_1) \overset{W}{\rightarrow} J_2 \overset{e}{\rightarrow} \dots \overset{e}{\rightarrow} e(J_n) \end{cases} \quad (49)$$

Where R_t: the process of decrypting the password by the rainbow tables of the BTFAT; G_0, F_0, J_0: possible passwords; $e(G_0), e(F_0), e(J_0)$: the encrypted passwords; W: a function to convert the encrypted password.

Corollary-3: A hacker can hack any wireless network by getting a handshake. Starting with the first stolen encrypted password (B_s), the hacker applies the functions W and e repeatedly, calculating a series of encrypted passwords and final passwords, until he reaches the encrypted password with twenty 0 in front of it. The hacker then searches for this last encrypted password in the table (encrypted password E) and identifies the corresponding password (password E).

Table 4 - Rainbow table

Password Q	Encrypted password Q
Password W	Encrypted password W
Password E	Encrypted password E
Password R	Encrypted password R

The hacker then applies the W and e functions again, starting with the identified password, continuing until one of the received encrypted passwords in the chain matches the stolen encrypted password:

$$Password\ E \rightarrow Encrypted\ password\ 1 \rightarrow Password\ 2 \rightarrow Encrypted\ password\ 2 \\ \rightarrow Password\ 3 \dots \rightarrow \cdots Password\ 33 \rightarrow Encrypted\ password\ 34 \begin{bmatrix} a\ match\ to\ encrypted \\ password\ B_s \end{bmatrix}$$

Where $Password\ E, Password\ 1,2,3 \dots$: intermediate and final password required for hacking Wi-Fi; $Encrypted\ password\ 1, 2, \dots$: encrypted password located in the handshake.

An encrypted password that matches (*Encrypted password 34*) will mean that the previous password (*Password 33*) from which it was obtained is associated with the stolen encrypted password. To set the first and last columns of the rainbow table, you need to perform a lot of calculations. They store only the data in these two columns, and by recalculating the chain, hackers can identify any password by its encrypted password located in the handshake.

Experimental results

This section contains the proposed BTFAT. To demonstrate the advantages of choosing this tool for calculating the values of such characteristics as reliability, efficiency, and time of user de-authorization during hacking of a wireless network, these data were also calculated for three other tools (Reaver, Wifite, Wireshark).

Network hacking requires the following components, which are described in Table 5.

Table 5 - Components for hacking Wi-Fi

Components	Version/The name of the system
Personal computer	x64
Operation system	Linux Kali 5.9.0
Wireless access point	D-linkDIR-615
Resolution	1920x1080 px
Processor	Intel(R) Core (TM) i7-8750H
Maker	Acer
RAM	2048 MB
Video memory	16 MB
HARD Disk	39,9 GB (/dev/sda1)
CPU MHz	2208.002
Cash size	9216 KB

Based on the results, the following metrics are measured.

- Effectiveness of packet capture
- Client de-authorization time
- Reliability
- Processing performance of sent requests.

A. Effectiveness of Packet-capture

The effectiveness of using a particular method when hacking Wi-Fi is calculated by the equation:

$$E = \frac{P}{T} \times 100\% \qquad (50)$$

Where *E:* the effectiveness of packet capture; *P:* the number of packets captured; *T:* the time taken for a packet capture.

The data for calculating the packet capture effectiveness is given in Table 3 and Table 4. When calculating the effectiveness, the number of captured packets is considered. Table 3 and Table 4 show the number of packets captured by various tools within 50 seconds. The largest number of packets in a time equal to 50 seconds was captured by the BTFAT (48.5), the smallest number of packets – by Wifite (41). Figure 4 shows the effectiveness of packet capture using the BTFAT, Reaver, Wifite, and Wireshark tools. Figure 4 shows that BTFAT (97%) has the highest effectiveness. Figure 4 also shows that the effectiveness of the BTFAT increases over time.

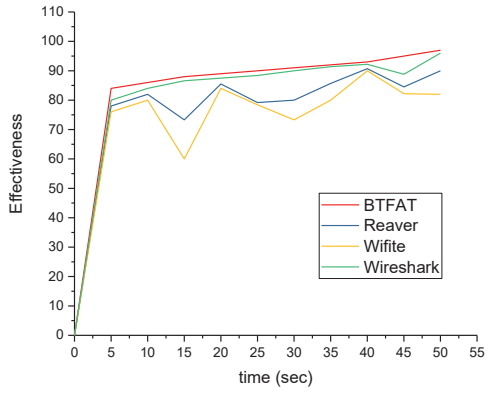

Figure 4 - The effectiveness of packet capture

B. Client de-authorization time

The client de-authorization time depends on the number of requests made by the hacker and the responses received from the client, as well as the speed of sending requests. The de-authorization time is calculated using the equation:

$$t = \frac{N_r \times N_a}{V_c} \quad (51)$$

Where t: the client de-authorization time; N_r: the number of requests; N_α: the responses received from the client; V_c: the speed of sending requests.

Data for calculating the de-authorization time are given in Table 5 and Table 6. Figure 5 shows the client de-authorization time for each tool. If the speed of sending requests is the same for all tools, then calculating the de-authorization time by the equation (51), it is noticeable that the de-authorization time increases with the passage of time and the requests sending. Figure 5 shows that the BTFAT sent 50 requests and the de-authorization time took 116.6 seconds. Thus, the BTFAT can complete the authorization process in a shorter time compared to other tools, which contributes to faster handshake establishment for password decryption.

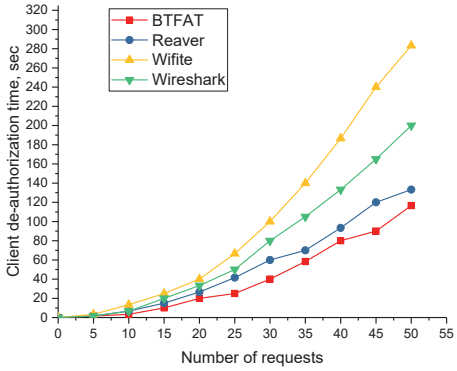

Figure 5 - Client de-authorization time

Figure 6 shows that for BTFAT, even with an increased number of requests, the pre-authorization time is minimal compared to other tools.

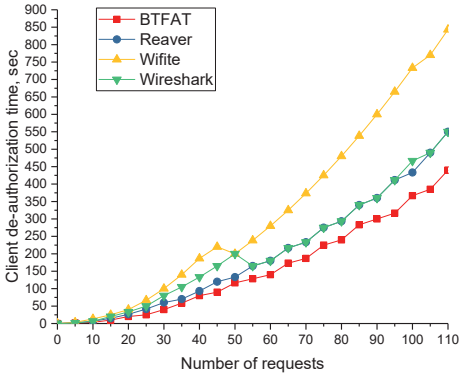

Figure 6 - Client de-authorization time

Figure 7 shows the relationship between the number of requests sent to the user and the number of responses received from the user. Figure 7 shows that the smallest number of responses received was accepted by the BTFAT (6). This means that the BTFAT requires less resources and time to intercept the handshake, as fewer requests are processed.

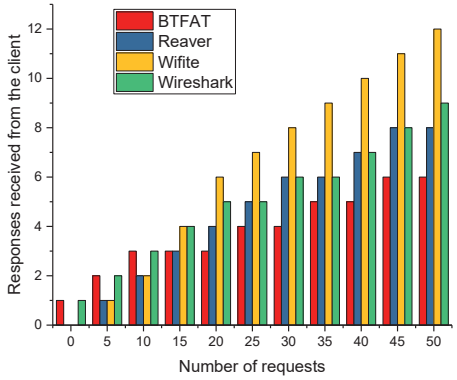

Figure 7 - Elements of the de-authorization process

Figure 8 shows the relationship between user requests and responses. The duration of a network hacker attack depends on the number of responses received as a result of requests sent by the hacker. The fewer responses received from the client and the user are de-authorized, the faster the user processes requests and the wireless network is hacked.

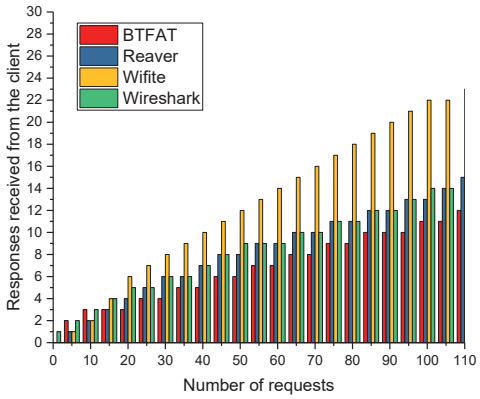

Figure 8 - Elements of the de-authorization process

During the de-authorization process, the time of this process depends on such elements as the number of requests and responses, and the speed of sending requests. Figure 9 shows the correlation between speed and time, as well as between the time and number of client responses. Figure 9 shows that the correlation values in the upper graph are less scattered, which means a higher correlation. In the lower graph, the values are more scattered, which means a high correlation. Table 9, showing the correlation coefficient of each element of the de-authorization process, demonstrates a 92% correlation between the time spent on client de-authorization and the number of responses received as a result of requests. This means that the de-authorization time is highly dependent on the number of responses received. The correlation between the speed of sent packets and the de-authorization time is 53%, and an average noticeable relationship is formed. This means that the de-authorization time is weakly dependent on the speed of sending packets.

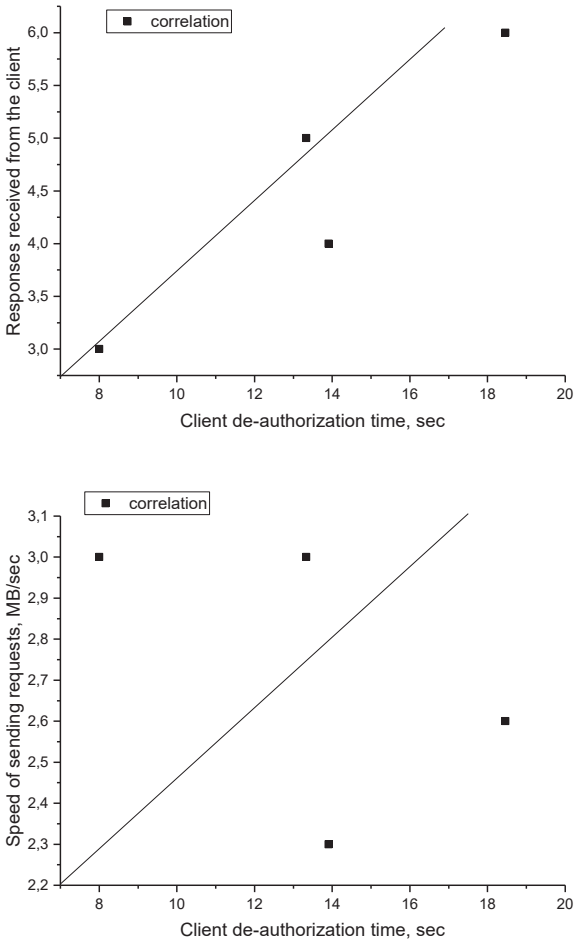

Figure 9 - Correlation dependence

C. Reliability

The reliability of Wi-Fi hacking is characterized by the probability of password decryption, which is determined by the following equation (52):

$$P_c(t) = \frac{N_0 - \sum n_i}{N_0} \qquad (52)$$

Where $P_c(t)$: reliability of Wi-Fi hacking; N_0: the number of initially captured packets; $\sum n_i$: the total number of requests.

Figure 10 presents the percentage of reliability of the network hacking process for each tool. Data for calculating the reliability of using each tool are shown in Table 10 and Table 11. Figure 10 shows that the BTFAT has the highest reliability (86%), and the Wifite tool has the lowest reliability (66%). Also, Figure 8 reveals that over time, the reliability of packet capture using the BTFAT remains higher than with other tools.

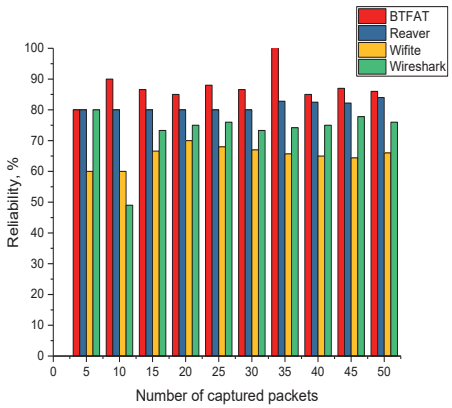

Figure 10 - Reliability of the network hacking process

D. Processing performance of sent requests

The processing performance of the requests sent to the user affects the time of the de-authorization process, as well as the process of handshake interception. Therefore, this parameter affects the total time of Wi-Fi hacking. The higher the performance, the faster a hacker can crack the Wi-Fi. The processing performance of the requests sent to the user is calculated using the equation:

$$P_p = \frac{\beta \times C}{\omega} \times 100\% \qquad (53)$$

Where P_p: the processing performance of sent requests to the user; β: the number of processed responses; C: the channel capacity; ω: the request processing time.

Figure 11 shows that the processing performance of requests sent by the BTFAT is stable compared to other tools and is equal to 85%. Also, the BTFAT has the highest performance, which contributes to the fastest Wi-Fi hacking. The data for calculating performance is described in Tables 12 and 13.

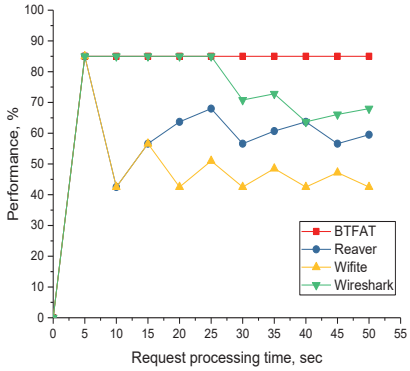

Figure 11 - The processing performance of the requests sent to the user

Discussion of results

The proposed BTFAT consists of three stages. The first stage is packet capture, the second is user de-authorization, and the last is Wi-Fi Blockchain-Featured Hacking Process. The advantages of using BTFAT is the use the features of Blockchain technology that capture the packets effectively, reduction of the user de-

authorization, and the reliability. The packet capture effectiveness is 97%, which is higher than that of the other tools. The user de-authorization time with the BTFAT is more effective as compared to other tools. This time is minimal when compared with other state-of-the-art tools. Thus, it proves that the BTFAT tool takes less time to perform de-authorization of the user, so the minimum amount of time is needed to intercept a handshake. The reliability of hacking a wireless network with BTFAT is 86%, which is the highest indicator. Table 3 shows the comparative analysis of the proposed BTFAT tool and other contending tools.

Another advantage of this tool is that BTFAT works with any wireless network adapters whose driver supports the monitoring mode. Also, the advantage of this tool is its extensive functionality. In addition to cracking WEP/WPA/WPA2 keys, BTFAT can decrypt intercepted traffic with a known key, analyze traffic, create a virtual tunneling interface, create encrypted packets for injection, provide techniques for attacking the client, remove WEP masking from PCAP files, store and manage lists of ESSIDs and passwords, calculate paired master keys, and open access to the wireless network card from other computers. However, this method of hacking Wi-Fi has disadvantages. The main disadvantages are the slow speed of password search and the lack of tables with pre-calculated hashes for password selection.

Table 3 - Comparative analysis of the proposed BTFAT, Reaver, Wifite and Wireshark tools

Name of tools	Effectiveness of packet capture	Client de-authorization time		Responses received from the client 55 Requests	Responses received from the client 110 Requests	Reliability Maximum captured 50 packets	Processing performance of sent requests Request processing 50 seconds
		55 Request	110 Request				
BTFAT	97%,	116.6	192	06	12	86%	85%
Reaver	87.4%	133.3	551.2	08	16	84%	59.5%
Wifite	74.3%	283.3	796	12	24	66%	47.2%
Wireshark	94.3%	200	552	9	18	76%	66.1%

Conclusion

This paper introduces a Blockchain-featured BTFAT for controlling the hacking of the wireless network. It also provides a detailed description of the wireless network hacking process. The Wi-Fi hacking process occurs in three phases. In the beginning, packets are captured by monitoring and saved to a file, then the user is de-authorized, and the handshake is recorded in a previously saved file. The last phase consists of Blockchain technology features, which are used for controlling the hacking process of the wireless network and decrypting the password. The advantage of the proposed BTFAT is that the BTFAT can hack networks that use Blockchain technology, given that networks with such technology have very high security. Another advantage is the speed of using this method, its efficiency, and reliability.

The expressions have been used to calculate the values of efficiency, reliability, and user de-authorization time, and a comparative analysis of several tools for hacking Wi-Fi was performed. The reliability of using the BTFAT is 86%, efficiency - 97%, the request processing performance time is 85%. Furthermore, the time of detecting the Wi-Fi-hacking is minimal compared to other existing state-of-the-art tools. These results show that the proposed BTFAT is the best choice for Wi-Fi-hacking prevention. In the future, we will model the pen-testing process with BTFAT for evaluating the wireless network security metrics.

REFERENCES

[1] Cisar, P., and S. Maravic Cisar. "Ethical hacking of wireless networks in kali Linux environment." Annals of the Faculty of Engineering Hunedoara 16.3 (2018): 181-186.

[2] Astudillo, Karina. Wireless Hacking 101. Babelcube Inc., 2017.

[3] Karagiannis, Konstantinos. "Hacking Blockchain." (2017).

[4] Venkatesh, V. G., et al. "System architecture for blockchain based transparency of supply chain social sustainability." Robotics and Computer-Integrated Manufacturing 63 (2020): 101896.

[5] Werbach, Kevin. The blockchain and the new architecture of trust. Mit Press, 2018.

[6] Sinha, Sanjib, Sanjib Sinha, and Karkal. Beginning Ethical Hacking with Kali Linux. Apress, 2018.

[7] Ansari, Juned Ahmed. Web penetration testing with Kali Linux. Packt Publishing Ltd, 2015.

[8] Guo, Rui. "Survey on WiFi infrastructure attacks." International Journal of Wireless and Mobile Computing 16.2 (2019): 97-101.

[9] Pimple, Nishant, et al. "Wireless Security—An Approach Towards Secured Wi-Fi Connectivity." 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS). IEEE, 2020.

[10] Noshad, Zainib, Nadeem Javaid, and Muhammad Imran. Analyzing and securing data using data science and blockchain in smart networks. Diss. MS thesis, COMSATS University Islamabad (CUI), Islamabad 44000, Pakistan, 2019.

[11] Swedan, AbedAlqader, et al. "Detection and prevention of malicious cryptocurrency mining on internet-connected devices." Proceedings of the 2nd International Conference on Future Networks and Distributed Systems. 2018.

[12] Kabanov, P. A., and Mikhail Sergeevich Sukhodoev. "Overview of hacking tools and protection of modern ICT devices." 14th International Forum on Strategic Technology (IFOST-2019), October 14-17, 2019, Tomsk, Russia:[proceedings].—Tomsk, 2019.. 2019.

[13] Goyal, Piyush, and Anurag Goyal. "Comparative study of two most popular packet sniffing tools-Tcpdump and Wireshark." 2017 9th International Conference on Computational Intelligence and Communication Networks (CICN). IEEE, 2017.

[14] Astudillo, Karina. Wireless Hacking 101. Babelcube Inc., 2017.

[15] Vance, William. Linux for Hackers: A Comprehensive Beginners Guide to the World of Hacking using Linux. joiningthedotstv, 2020.

[16] Таганов, П. А. "Исследование алгоритма атаки на беспроводную сеть Wi-Fi." Организатор конференции. 2018.

[17] Parasram, Shiva VN, et al. Kali Linux 2018: Assuring Security by Penetration Testing: Unleash the full potential of Kali Linux 2018, now with updated tools. Packt Publishing Ltd, 2018.

[18] Baloch, Rafay. Ethical hacking and penetration testing guide. CRC Press, 2017.

[19] Carranza, Aparicio, et al. "Automated Wireless Network Penetration Testing Using Wifite and Reaver." Global Partnerships for Development and Engineering Education: Proceedings of the 15th LACCEI International Multi-Conference for Engineering, Education and Technology, July 19-21, 2017, Boca Raton, FL, United States. Latin American and Caribbean Consortium of Engineering Institutions, 2017.

[20] Carranza, Aparicio, et al. "Automated Wireless Network Penetration Testing Using Wifite and Reaver." Global Partnerships for Development and Engineering Education: Proceedings of the 15th LACCEI International Multi-Conference for Engineering, Education and Technology, July 19-21, 2017, Boca Raton, FL, United States. Latin American and Caribbean Consortium of Engineering Institutions, 2017.

[21] Martin, Alexander, Basiru Mohammed, and Rajkumar Ramadhin. "WEP VS WPA2 Encryptions." (2019).

[22] Alassouli, Hidaia Mahmood. Hacking of Computer Networks. Dr. Hidaia Mahmood Alassouli, 2020.

[23] Al Neyadi, Eiman, et al. "Discovering Public Wi-Fi Vulnerabilities Using Raspberry pi and Kali Linux." 2020 12th Annual Undergraduate Research Conference on Applied Computing (URC). IEEE, 2020.

[24] Pimple, Nishant, Tejashree Salunke, Utkarsha Pawar, and Janhavi Sangoi. "Wireless Security—An Approach Towards Secured Wi-Fi Connectivity." In 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), pp. 872-876. IEEE, 2020.

[25] Pandikumar, T., and Mohammed Ali Yesuf. "Wi-Fi Security and Test Bed Implementation for WEP and WPA Cracking." International Journal of Engineering Science 13571 (2017).

[26] Sharma, Himanshu. Kali Linux-An Ethical Hacker's Cookbook: Practical recipes that combine strategies, attacks, and tools for advanced penetration testing. Packt Publishing Ltd, 2019.

[27] Bullock, Jessey, and Jeff T. Parker. Wireshark for Security Professionals: Using Wireshark and the Metasploit Framework. John Wiley & Sons, 2017.

[28] Li, Lei, Zhigang Li, Hossain Shahriar, Rebecca Rutherfoord, Svetana Peltsverger, and Dawn Tatum. "Ethical Hacking: Network Security and Penetration Testing." (2018).

[29] Sinha, Sanjib. "Hashes and Passwords." Beginning Ethical Hacking with Kali Linux. Apress, Berkeley, CA, 2018. 323-345.

[30] Santo Orcero, David. Kali Linux. Grupo Editorial RA-MA, 2018.

[31] Sinha, Sanjib. "MAC Address." Beginning Ethical Hacking with Python. Apress, Berkeley, CA, 2017. 191-194.

Разак А., Әділ А.Ж., Аманжолова С.Т.

Блокчейн технологиясына негізделген Wi-Fi хакерін анықтаудың жаңа құралы

Аңдатпа. Wi-Fi бизнес, білім беру, өнеркәсіп және т.б. көптеген салаларда маңызды рөл атқарады, екінші жағынан, Wi-Fi осалдықтары пайдаланушылардың мәліметтерінің құпиялылығына зиян келтіреді, егер осалдықтар дұрыс өңделмесе. Кейбір хакерлер бұзу процесіне әкелетін Wi-Fi осалдығын пайдалану үшін Linux құралын пайдаланады. Бұл мақалада Wi-Fi желісінің қауіпсіздігін жақсарту үшін Blockchain Technology-Featured Novel Air-Cracking tool (BTFAT) ұсынылған. Құрал құнды функциялардан тұрады (мысалы, бақылау, сканерлеу, бұзу және тестілеу). Бұл функциялар желінің осалдықтарын анықтауға көмектеседі, BTFAT C тілінде бағдарламаланған. Эксперимент нәтижелеріне сүйене отырып, BTFAT басқа қолданыстағы әдістермен салыстырғанда жоғары өнімділікті қамтамасыз етеді.

Кілт сөздер: Wi-Fi, осалдық, BTFAT, құпиялылық, сенімділік, тестілеу, Blockchain технологиясы.

Разак А., Әділ А.Ж., Аманжолова С.Т.

Новый инструмент для обнаружения взлома Wi-Fi на основе технологии блокчейн

Аннотация. Wi-Fi играет важную роль во многих областях, таких как бизнес, образование, промышленность и т. д. С другой стороны, уязвимости Wi-Fi наносят ущерб конфиденциальности данных пользователей, если уязвимости не обрабатываются должным образом. Некоторые хакеры используют инструмент Linux для использования уязвимости Wi-Fi, которая приводит к процессу взлома. В этой статье представлен новый инструмент Blockchain Technology-Featured Novel Air-Cracking tool (BTFAT) для улучшения безопасности сети Wi-Fi. Инструмент состоит из ценных функций (например, мониторинг, сканирование, взлом и тестирование). Эти функции помогают обнаружить уязвимости сети, запрограммирован на языке C. Основываясь на результатах эксперимента, BTFAT обеспечивает более высокую производительность по сравнению с другими существующими методами.

Ключевые слова: Wi-Fi, уязвимость, BTFAT, конфиденциальность, надежность, тестирование, технология блокчейн.

Авторлар туралы мәлімет:

Абдул Разак, «Киберқауіпсіздік» кафедрасының профессоры, Халықаралық ақпараттық технологиялар университеті.

Әділ Алтынай Жанарбекқызы, «Компьютерлік инженерия» кафедрасының магистранты, Халықаралық ақпараттық технологиялар университеті.

Аманжолова Сауле Токсановна, «Киберқауіпсіздік» кафедрасының меңгерушісі, Халықаралық ақпараттық технологиялар университеті.

Валиев Бахытжан Бауржанович, «Компьютерлік инженерия» кафедрасының магистранты, Халықаралық ақпараттық технологиялар университеті.

Сведения об авторах:

Абдул Разак, профессор кафедры «Кибербезопасность», Международный университет информационных технологий.

Әділ Алтынай Жанарбекқызы, магистрант кафедры «Компьютерная инженерия», Международный университет информационных технологий.

Аманжолова Сауле Токсановна, заведующая кафедрой «Кибербезопасность», Международный университет информационных технологий.

Валиев Бахытжан Бауржанович, магистрант кафедры «Компьютерная инженерия», Международный университет информационных технологий.

About the authors:

Razaque A., Professor, Department of Cybersecurity, International Information Technology University.

Adil A.Zh., Master student, Department of Computer Engineering, International Information Technology University.

Amanzholova S. T., Head of the Department of Cybersecurity, International Information Technology University.

Valiyev B.B., Master student, Department of Computer Engineering, International Information Technology University.

INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGIES

МЕЖДУНАРОДНЫЙ ЖУРНАЛ ИНФОРМАЦИОННЫХ И КОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

ХАЛЫҚАРАЛЫҚ АҚПАРАТТЫҚ ЖӘНЕ КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАР ЖУРНАЛЫ

Ответственный за выпуск	Есбергенов Досым Бектенович
Редакторы	Медведев Евгений Юрьевич
Компьютерная верстка и дизайн	Жадыранова Гульнур Даутбековна

Подписано в печать 15.12.2021 г.
Тираж 500 экз. Формат 60x84 1/16. Бумага тип.
Уч.-изд.л. 6.5. Заказ №170

Издание Международный университет информационных технологий
Издательский центр КБТУ, Алматы, ул. Толе би, 59