Computers, Materials & Continua
DOI:10.32604/cmc.2021.013951
Article

Tech Science Press

An Improved Dictionary Cracking Scheme Based on Multiple GPUs for Wi-Fi Network

Majdi K. Qabalin[1], Zaid A. Arida[2], Omar A. Saraereh[3], Falin Wu[4,*], Imran Khan[5], Peerapong Uthansakul[6] and Moath Alsafasfeh[7]

[1]Department of Computer Science, Princess Sumaya University for Technology, Amman, 13133, Jordan
[2]School of Computing and Informatics, Al Hussein Technical University, Amman, 11831, Jordan
[3]Department of Electrical Engineering, Hashemite University, Zarqa, 13133, Jordan
[4]School of Instrumentation and Optoelectronic Engineering, Beihang University, Beijing, 100191, China
[5]Department of Electrical Engineering, University of Engineering and Technology, Peshawar, Pakistan
[6]School of Telecommunication Engineering, Suranaree University of Technology, Nakhon Ratchasima, 30000, Thailand
[7]Department of Computer Engineering, Al-Hussein Bin Talal University, Ma'an, Jordan
[*]Corresponding Author: Falin Wu. Email: falin.wu@buaa.edu.cn
Received: 26 August 2020; Accepted: 19 October 2020

Abstract: The Internet has penetrated all aspects of human society and has promoted social progress. Cyber-crimes in many forms are commonplace and are dangerous to society and national security. Cybersecurity has become a major concern for citizens and governments. The Internet functions and software applications play a vital role in cybersecurity research and practice. Most of the cyber-attacks are based on exploits in system or application software. It is of utmost urgency to investigate software security problems. The demand for Wi-Fi applications is proliferating but the security problem is growing, requiring an optimal solution from researchers. To overcome the shortcomings of the wired equivalent privacy (WEP) algorithm, the existing literature proposed security schemes for Wi-Fi protected access (WPA)/WPA2. However, in practical applications, the WPA/WPA2 scheme still has some weaknesses that attackers exploit. To destroy a WPA/WPA2 security, it is necessary to get a PSK pre-shared key in pre-shared key mode, or an MSK master session key in the authentication mode. Brute-force cracking attacks can get a phase-shift keying (PSK) or a minimum shift keying (MSK). In real-world applications, many wireless local area networks (LANs) use the pre-shared key mode. Therefore, brute-force cracking of WPA/WPA2-PSK is important in that context. This article proposes a new mechanism to crack the Wi-Fi password using a graphical processing unit (GPU) and enhances the efficiency through parallel computing of multiple GPU chips. Experimental results show that the proposed algorithm is effective and provides a procedure to enhance the security of Wi-Fi networks.

Keywords: Networks; password; cybersecurity; password cracking mechanism

1 Introduction

Wireless Fidelity (Wi-Fi) is a technology that connects electronic devices to a Wireless Local Area Network (WLAN), usually using 2.4 GHz or 5 GHz radio frequency bands [1,2]. Wi-Fi signals are typically password-protected, but it can also be turned on so that all devices in the range are connected. This technology improves the interoperability of wireless network devices based on the IEEE 802.11 standard [3–6]. In 2014, the number of public Wi-Fi hotspots exceeded 50 million which is an 80% increase over 2013 [7–10]. Among them, China ranks fourth with more than 4.91 million public hotspots [11–13]. With numerous wireless hotspots applications, it's inevitable to study the dangers of security to ensure the safety of the subscriber.

Some of the hidden threats for Wi-Fi are as follows. Service Set Identifier (SSID) broadcasts, which are very easy to discover. Wireless routers have a simple WPS link function in addition to the account login password which can be easily changed by wireless router administrators and regular users. For more intuitive Wi-Fi password security issues, there are also many corresponding tools. One is software and hardware cracking under windows or Linux environments. These are not only deployed for WEP cracking, but also for routing PIN code-cracking [14]. At present, the principle of using multiple terminal applications such as the master key is to share the Wi-Fi account password that the user has in the background to the server and then retrieve the corresponding entry from the server database. In addition, there are different types of the terminal for cracking and the effect is very limited.

This article proposed a new method that uses the efficient parallel computing power of GPU and the common user characteristics to produce a various combination of dictionaries to crack the passwords under the existing security mechanisms in Wi-Fi networks. The experimental analysis is performed for different important network parameters.

The rest of the article is organized as follows. In Section 2, the principles of a security protocol are described. In Section 3, the difficulties faced by the current cracking method is discussed. In Section 4, the dictionary-based cracking model is described. Section 5 provides the analysis of the proposed scheme while Section 6 gives the conclusion.

2 Principles of WEP/WPA/WPA2 Security Protocol

The Wi-Fi password security mechanism consists of three categories which are WEP, WPA, and WPA2 [15]. Among them, the WEP security has proven to be unreliable and the major flaw lies in the insecure initialization of vector and the CRC algorithm [16]. Therefore, the Wi-Fi came in. WPA/WPA2 standards and protocols protect the security of wireless networks. They are divided into enterprise-level, WAP-enterprise and personal-level WPA-PSK. The enterprise-level uses RADIUS authentication which is a network protocol for remote user authentication, authorization and accounting [17]. It is safe and will not be mentioned in this article. Personal-level WPA-PSK using Pre-Shared Key technology and does not require a dedicated authentication server. This technology only requires each WLAN node to enter a key in advance. The WPA/WPA2 is composed of three parts which are encryption, authentication and message integrity verification to form a complete security solution. The WPA protocol uses the Temporary Key Integration Protocol (TKIP) algorithm, Message Integrity Coding (MIC) and Counter Mode Cipher Block Chaining Message Authentication Code Protocol (CCMP). The working principle of the WPA/WPA2-PSK mode-based handshake mechanism is a verification and authentication process for MIC [18]. The process includes the following parts which are Master Session Key (MSK), Pairwise Master Key (PMK), Pairwise Transient Key (PTK), Group Master Key (GMK)

and Group Transient Key (GTK). Among them, the PTK protocol contains the following three parts which are Key Confirmation Key (KCK), Key Encryption Key (KEK) and Temporal Key (TK). The KCK calculates the integrity of the key generation message, KEK encrypts the key generation message and TK which encrypts the data. The PMK comes from MSK. The five necessary elements for MSK to generate PTK are PMK, ANonce (AN), SNonce (SN), Authenticate MAC (AA) and Supplicant MAC (SA). The two send random numbers for the wireless access point (AP) and the client STA. The process is shown in Fig. 1.

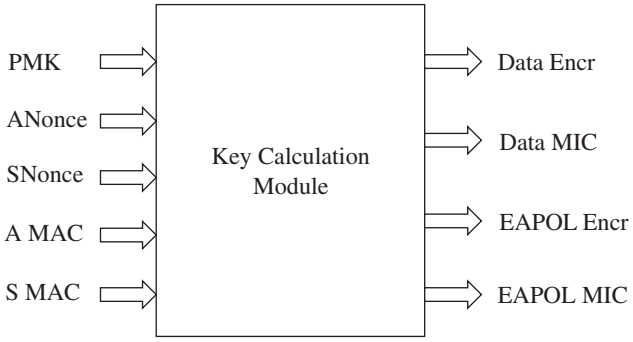

Figure 1: Generation of PTK from PMK

The output of Fig. 1 consists of four parts, where Data Encr, Data MIC combine, TK, EAPOL Encr/MIC, KEK, and KCK, respectively. The process is as follows. It uses the AP's SSID and passphrases, adopts the Pdkdf2 algorithm to generate PMK where passphrase and SSID are deployed. *A priori* condition of authenticator and supplicant before authentication in WPA-PSK is PSK = PMK = pdkdf2_SHA1 (passphrase, SSID, SSID length, 4096). Then the SHA1_PRF algorithm is used to generate PTK from PMK, SA, AA, SN and AN. That is, PTK = SHA1_PRF (PMK, Len (PMK), "Pairwise key expansion," MIN (AA, SA) ‖MAX (AA, SA) ‖MIN (AN, SN) ‖MAX (AN, SN)). Then, GMK gets GTK = PRF-X (GMK, AA‖GN) through expansion, where GN is the nonce generated by the authenticator and AA is the MAC address of the authenticator.

In the entire process, a 4-way handshake is required before updating the paired key. The paired key is used to encrypt the unicast frames, the communication frame between the AP and the workstation. The 4-way handshake is to complete the exchange of only one paired key so that both parties can learn about it.

The 4-way handshake messages are all based on EAPOL-Key and its structure is shown in Tab. 1.

The focus of this article is to initially compute the PMK with a password. Therefore, we can know whether the password is correct by the second handshake. The specific process is shown in Fig. 2.

The cracking process is summarized in the following steps:

a) Passively check the communication between AP and STA to receive 4 packets, handshake and AP SSID.
b) Perform exhaustive keying offline and use SSID to calculate PMK.
c) Analyze handshake packets 1 and 2 to get AN, SN, AA, SA, and use the obtained PMK to calculate the PTK.

d) Use the obtained KCK of the first 128 bits of PTK to calculate the MIC value of handshake packet 2 and compare it with the MIC value of the captured handshake packet 2. If they are the same, the key is correct; otherwise, try the next key.

Table 1: EAPOL-Key descriptor structure

Protocol version 1B	Packet type 1B	Packet body length 2B
Descriptor types 1B		
Key information 2B		Key Length 2B
Key replay counter 8B		
Key nonce 32B		
EAPOL-key IV 16B		
Key RSC 8B		
Key Flag 8B		
Key MIC 16B		
Key data length 2B		Key Data 0 ~ NB

3 Current Cracking Difficulties and Cracking Method Design

Compared to the WEP protocol, the WPA/WPA2 protocol improves security from encryption algorithms to data integrity verification algorithms. To decipher a Wi-Fi password from a protocol point of view, no serious flaws can be exploited but we can still decrypt it from the perspective of guessing the user's password. One way to decipher a guessed password is to first use a packet capture tool to capture a handshake data packet in the above WPA/WPA2 handshake process. In these handshake steps, a mathematical operation is included, namely the Hash algorithm. It is an irreversible operation and the result cannot infer the original number. The results obtained by the Hash algorithm for two different values are usually different. In the Wi-Fi handshake process, the existing PSK dictionary library plus SSID are used to get 64 bytes of PMK through calculations and compare it with the MIC value obtained by capturing the packet to know the Wi-Fi password, which is commonly known as dictionary cracking.

However, the disadvantage of this method is that the PMK has a relatively large computational overhead, which usually takes a long time to crack using the CPU [19]. With the continuous development of computer hardware and the improvement of the single computing power of GPU, we consider the combination of CPU and GPU parallel display chip to improve the efficiency of PMK Wi-Fi dictionary computation and perform comparison [20].

The GPUs have advantages over CPUs in processing power and memory capacity. Because of the high parallelism of graphics, the GPU can increase processing power and memory capacity by increasing the was the effect, memory control unit integer calculations, fragmented logic judgments, and CPU decimal operations are performed by different arithmetic units with floating-point accelerators. Therefore, the CPU will have different performance when faced with different computing tasks. In contrast, the GPU performs integer and floating-point calculations

using the same arithmetic unit, so the GPU's integer computing capability is the same as the floating-point capability.

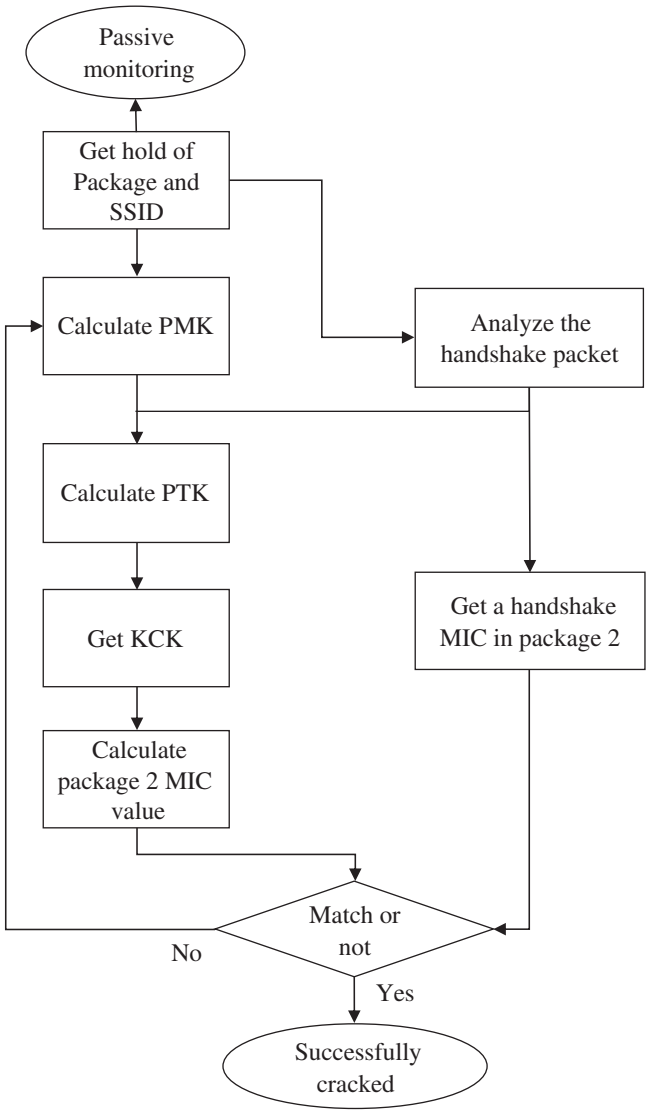

Figure 2: Flowchart of the cracking process

At present, mainstream GPUs have adopted unified architecture units. With a powerful lineup of programmable stream processors, GPUs surpass the capability of CPUs in single-precision floating-point operations. For example, the Intel Core i7 965 processor has only one-tenth of the floating-point computing power in the default configuration of the NVIDIA GeForce GTX 280. In addition, the advantage of GPU computing over CPU lies in video memory. The measured capacity of the desktop product DDR3-1333 is mostly around 20 GB/s, while the GTX280 using high-frequency GDDR3 memory has a memory controller that can support 512 bits wide and the memory capacity can reach about 140 GB/s. Such ultra-high capacity ensures stable operation of huge floating-point computing capabilities and enables efficient computation of intensive data.

A related architecture platform that can call GPU resources is Compute Unified Devices Architecture (CUDA). The CUDA architecture was launched by NVIDIA in 2006 [21]. As the current display chip has high programmability, so the memory capacity and the number of execution units of the display chip have also been greatly increased. To enable the display chip to compute data, the General Purpose GPU (GPGPU) was developed and CUDA is the model of NVIDIA's GPGPU. As far as GPU products are concerned, the products from ATI and NVIDIA are mainly popular on the market. In view of the deployment characteristics and the existing hardware resources, this article uses NVIDIA's display chip products and the corresponding CUDA architecture. The ATI display chip was developed by OpenCL technology as its product stream architecture. The company's GPUs can be used for general-purpose computing. The previously parallel processing used only CPUs which can be effectively expanded from large clusters to the graphics card, reducing the cost of computing applications and offering outstanding performance.

Therefore, this article proposes an improved dictionary cracking scheme that uses multiple CPUs and GPUs to combine parallel operations and improve the efficiency of dictionary comparison cracking [22]. In the specific calculation process, this article uses the Pyrit tool for calculation [23] which allows creating massive databases to pre-calculate the realization of WPA/WPA2-PSK in the authentication phase of space exchange time, which can utilize the computing power of multi-core CPUs and other platforms such as NVIDIA's CUDA. It provides multiple functions such as analysis, attack batch, attack cow patty, attack db, attack pass through, benchmark, and import password.

If we only rely on CPUGPU hybrid parallel computing capability to speed up computing efficiency, the running efficiency of a random dictionary remains astronomical numbers, which are practically insignificant. Therefore, while using hardware acceleration, this article also imposes certain prior conditions on the combination of dictionary packages. Researchers commonly used dictionary combinations and generate the corresponding dictionary package files to reduce the amount of hardware traversal, comparison operations, and improve the efficiency of guess passwords.

4 Dictionary Combination in Cracking Model

According to calculations, the number of records in the dictionary package file with 1 to 6 digits plus uppercase and lowercase English letters is about 57 billion, which occupies about 430 GB of hard disk memory. The number of random combination dictionary packets of 1 to 6 pure numbers is about 1.1 million, which occupies about 8.4 MB of memory. The number of rows in the dictionary with a random combination of 1 to 6 uppercase and lowercase English letters is about 20 billion, accounting for about 149 GB of memory. If we add a special symbol that is commonly used for random sorting, there will be more than 500 billion combinations, which is about 4 TB memory. We can see that one more optional combination for each digit of the password will produce much more output results. The results are shown in Figs. 3 and 4.

Similarly, with the same choice for each digit, one more digit in the password will increase the number of dictionary combinations by at least an order of magnitude. For example, there are 1 to 6 digits of a) pure numbers; b) uppercase and lowercase letters; c) numbers plus uppercase and lowercase letters; d) numbers plus uppercase and lowercase letters plus common special symbols. When the number of password digits increases from 1 to 7 digits, the number of lines in the dictionary combination is 1–100, over 10,000 items, over 1 trillion items, over 35,000 billion items, and over 48 trillion items, respectively. When the number of password digits increases from 1 to 8 digits, the number of dictionary combinations is over 100 million, 54 trillion, 220 trillion and

4,350 trillion, respectively. We can see that when the password combination contains alphanumeric characters and commonly used special symbols, it will consume an enormous amount of time and resources to traverse only a random password combination of 1 to 8 digits.

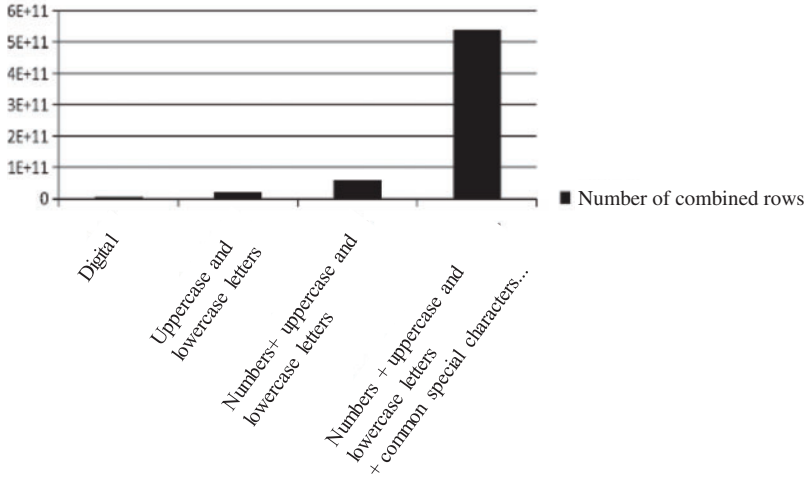

Figure 3: Variation of the number of combined rows with a combination

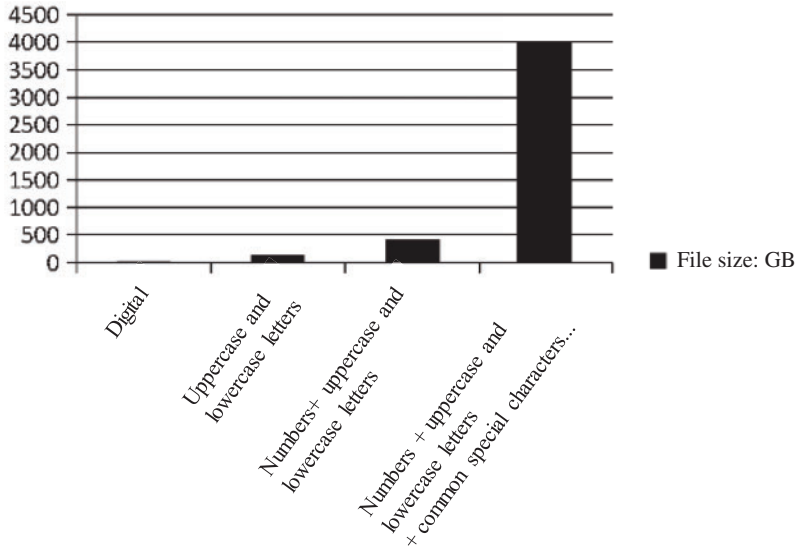

Figure 4: Variation in file size changes with a combination

This article believes that in practical use, the vast majority of users share a meaningful password. Therefore, to reduce the working volume of dictionary files, some common dictionary sets are offered to reduce the low hit rate dictionary comparison list.

The generated dictionary combination files with certain rules are: 1 to 6 digits can be repeated in a random combination of uppercase letters, a random combination of multiple lowercase

letters, and 1 to 8 repeatable numbers. Also, there are common 3000 English words, 1111 English names, 2371 English place names and 1,808 weak passwords that are commonly used.

Also, there is multi-format dates such as 19000101–20200101. Taking June 1, 1994, as an example, the formats include: a) the year, month, day (19940601); b) Year, month, day (940601); c) Year, month, day (9461); d) Year, month, day (1994.06.01); e) Year Month Day (1994-06-01); f) Year Month Day (1994/06/01); g) Year Month (199406), h) Year Day (199401), i) Month Day (0601); j) Month Day Year (06011994); k) Month Day Year (060194); l) month day year (06-01-1994); m) month day year (06/01/1994); n) day month year (01061994); o) Sun Moon Year (010694), etc. There are also several combinations not listed here due to limited space.

The pinyin of Chinese name is mainly based on the hundred family surnames, commonly known as Bai Jia Xing (including 472 characters with 3000 + common Chinese characters), the list of Bai Jia Xing is Zhao Qian Sun Li, Zhou Wu Zheng Wang, Feng Chen Chu Wei, Jiang Shen Han Yang, Zhu Qin You Xu, He Lu Shi Zhang, Kong Cao Yan Hua, Jin Wei Tao Jiang, etc. In addition, some combinations of Chinese names are given by Chinese single-character surname (lowercase and uppercase), Chinese three-character name (initial uppercase), Chinese double-character name (initial uppercase), and Chinese 1–3-character name (all lowercase, all uppercase and initial uppercase).

Regarding phone number, the example of the Shanghai area code is 021 + 8 random fixed-line number. In addition, according to the existing mobile number range, a mobile number dictionary library for all parts of the mainland includes China Mobile, China Unicom, Telecom and virtual operator numbers is generated which covers Shanghai, Beijing, Guangdong, Yunnan, Inner Mongolia, Sichuan, Tianjin, Ningxia, Shandong, Shanxi, Guangxi, Xinjiang, Jiangxi, Hebei, Henan, Zhejiang, Hainan and other parts of the country.

It also includes matching Chinese names and dates, common English words and dates, common mailbox suffixes and Chinese names, common English words and common dates combined, and about 160 GB of dictionary files. But these combinations are not enough to represent all the commonly used password combinations and it is necessary to constantly add common combinations. For example, the Chinese name and the date are combined with the mailbox suffix, Chinese name and English word combination. It also contains more than 600 MB of known aggregation files obtained from the Internet, with more than 670,000 combinations. We can summarize more common combination methods from this entry and generate the corresponding dictionary combination file, which improves the efficiency of comparing dictionary files with multiple parallel CPUGPUs.

With previous knowledge, the dictionary comparison range should be minimized as much as possible to improve the performance of dictionary packages in password cracking. The performance comparison of several groups of constrained dictionary running and complete exhaustion running is shown in Figs. 5–9. As shown in Fig. 5, the 8-digit date combinations from 1970 to 1990 is compared with the number of 8-digit random number combinations. Fig. 6 shows the comparison of shanghai fixed-line + Chinese first letter combination and 9–11 random number combination (lowercase only). Fig. 7 shows the comparison of Chinese name pinyin combinations and 6–12 random letter combinations (lowercase only). Fig. 8 shows the comparison of Chinese initials + Shanghai mobile number combination and 12 to 14 digits random number and letter combination (only lower case). Fig. 9 shows the combination of English name + date combinations and 10–17 random numbers and letters combinations (lower case only).

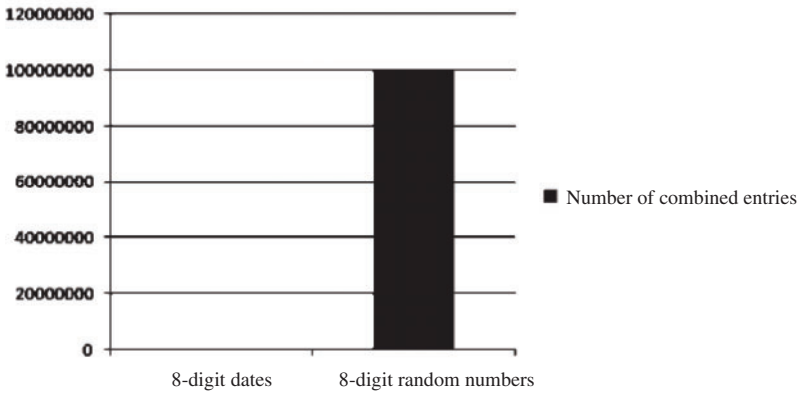

Figure 5: Comparison of 8-digit dates and 8-digit random numbers from 1970 to 1990

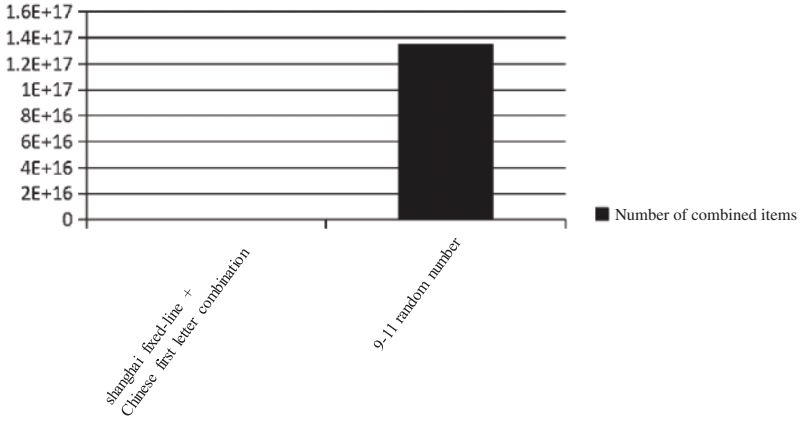

Figure 6: Comparison of shanghai fixed-line + Chinese first letter combinations and 9–11 random number combinations

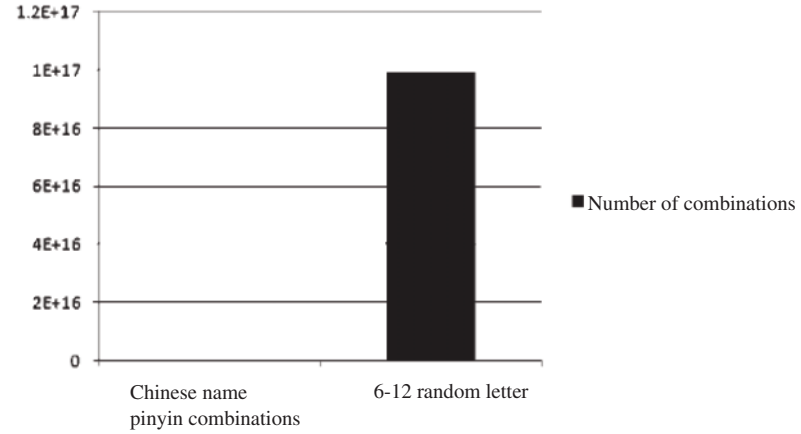

Figure 7: Comparison of Chinese name pinyin combinations and 6–12 random letter combinations

We can see from the sample data that the number of dictionary entries under the constraint is much less than the number of random numbers and letters, usually several orders of magnitude. Every time the password length increases by one digit, there is one more choice for the optional combination of each digit in the password which will further expand the number of constrained dictionary combination entries and the number of random combination dictionary entries. There is an enormous difference in the amount of computational time between the constrained dictionary and the random combination dictionary. In the actual operation process, avoiding meaningless or low hit probability calculations is an important way to improve efficiency.

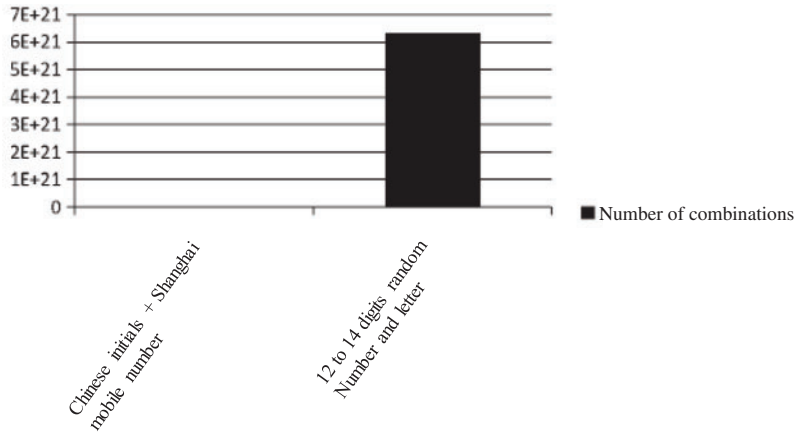

Figure 8: Comparison of Chinese initials + Shanghai mobile number combinations and 12 to 14 digits random number and letter combinations

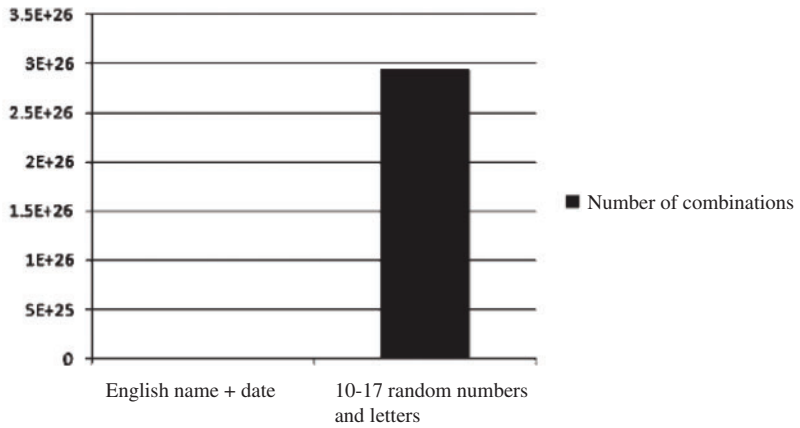

Figure 9: Comparison of English name + date combinations and 10–17 random numbers and letters combinations

5 Analysis of Hardware Acceleration Performance in the Crack Model

Based on the above analysis, characteristics of the algorithm, architecture, and related tools, the experimental process and the steps involved in this article are packet capture, handshake via

wireless network card and then run web service to send handshake packets to Servers using hybrid parallel processing capability of Multi-CPU-GPU for calculations and returns results.

The server hardware configuration is installed with two 16-core Intel Xeon E5-2630 v3@2.40 GHz CPU chips, 16 GB × 12 memory, 192 GB, and four GeForce GTX 1080 graphics chips, each with 8G GDDR5X video memory. We use the CentOS system.

After configuring the base server environment, we can set the specific CPU and GPU resources triggered by Pyrit. The settings interface is shown in Fig. 10. The limit_ncpus is the number of CPU cores to call and use_CUDA is the setting whether to call GPU resources or not.

```
default_storage = file://
limit_ncpus = 32
rpc_announce = true
rpc_announce_broadcast = false
rpc_knownclients =
rpc_server = false
use_CUDA = true
use_OpenCL = false
workunit_size = 75001
```

Figure 10: The Pyrit parameter settings

After setup, the first experiment is to use single-core CPU processing power. The running dictionary packet rate of a single-core pure CPU is about 400 + PMKs/s. After the long run of the package gets stable, the speed increases approximately 20% to 500 + PMKs/s. The results are shown in Figs. 11 and 12 where the abscissa is the serial number of each CPU core and the ordinate is the packet running rate of each CPU core in PMKs/s.

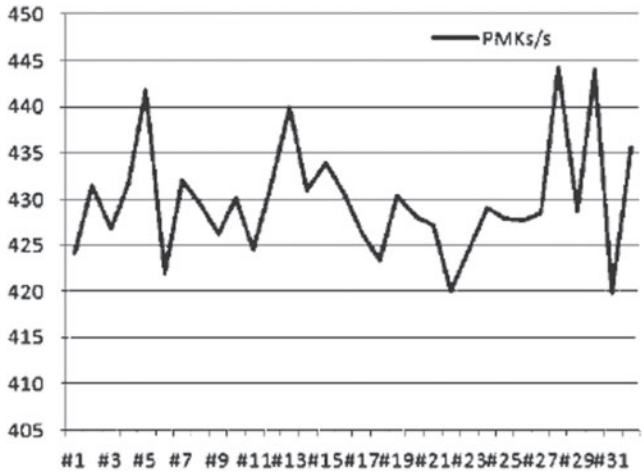

Figure 11: Single-core CPU packet rate

If cow patty is used for computation (only computational methods that run the CPU), the speed will not differ much from Pyrit's single-core CPU. If we set it to a pure GPU for running, the CPU packet computation will be disabled and the GPU rate is generally maintained at around 80000 PMKs/s by typing the command pyrite-r test.pcap-i dics.txt attack_passthrough. The

performance comparison is shown in Fig. 13 where the abscissa is the single-core CPU and the serial number of each GPU core, and the ordinate is the running speed of each core in PMKs/s.

Fig. 14 shows the variation of calculation speed over time. The experimental results showed that the dual-core CPU combined with the 4-core GPU gradually accelerated to compute larger dictionary packages and eventually remained stable at very high values after a long time (usually after 1 to 2 min). The abscissa is the cumulative time of the model running in seconds and the ordinate is the running speed of the model in PMKs/s.

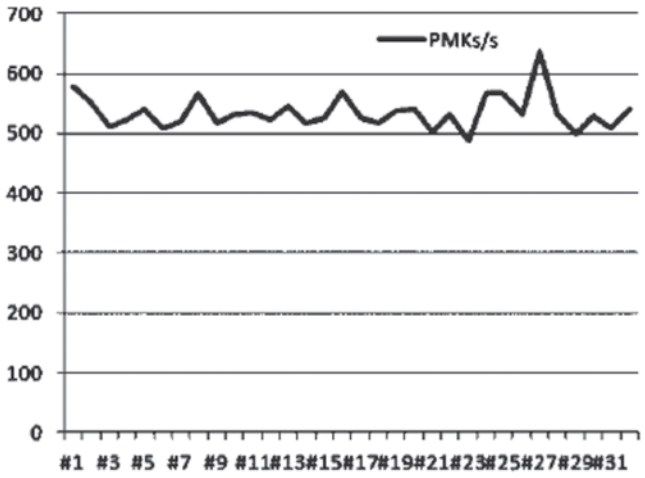

Figure 12: Long-term running single-core CPU package rate

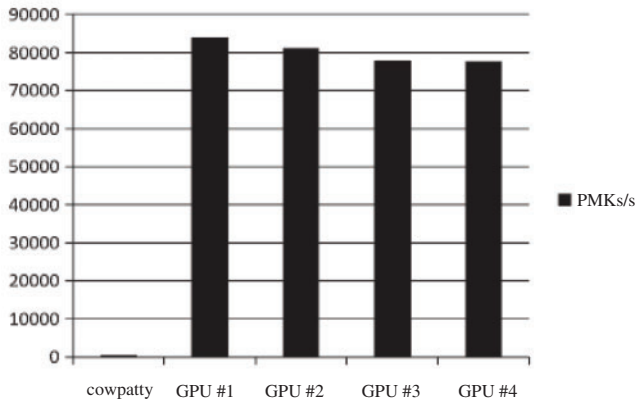

Figure 13: Single-core performance comparison

After generating many common dictionary files, the combination of CUDA+Pyrit is used to test various dictionary packages of different sizes. The results are shown in Fig. 15. The abscissa is the dictionary package of files of different sizes and the ordinate is the package running rate of each mode in PMKs/s.

It can be seen from the experimental results that it is too late to call the GPU resources when the dictionary package is small due to the powerful computer hardware nowadays. The limited

dictionary package data has been digested by powerful CPU computing performance. Therefore, the CPU responds to small data packages and the speed is better than GPU resource usage. When the amount of data exceeds approximately 2 MB, the CPU speed increases as processing time expands but it eventually stabilizes at 14,000 + PMKs/s. On the other hand, the overall performance of hybrid GPU and CPU-GPU computation is basically the same except for small data packets. The pure GPU performance is slightly higher than that of hybrid models, which is probably because of response time which is time the CPU-GPU coordination is eliminated in pure GPU mode. However, as the retrieval time of large dictionary data packets increases, the computation performance of GPU resource calls is greatly improved, and eventually stabilize around 120000 + PMKs/s which is about 10 times of pure CPU mode. The rates fluctuate slightly because of environmental influences.

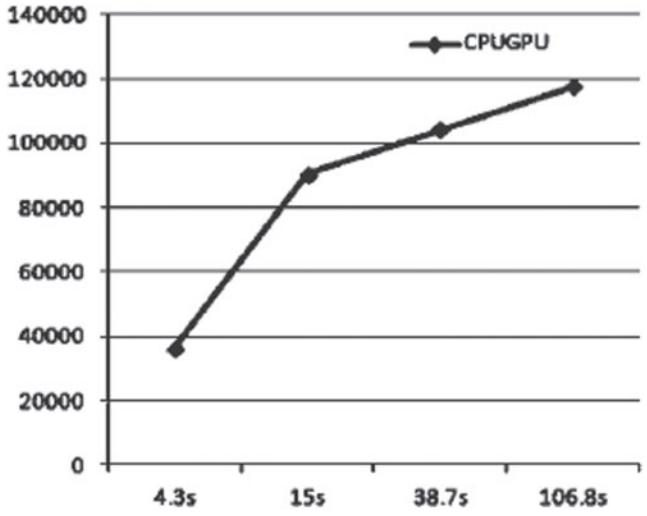

Figure 14: Variation of calculation speed over time

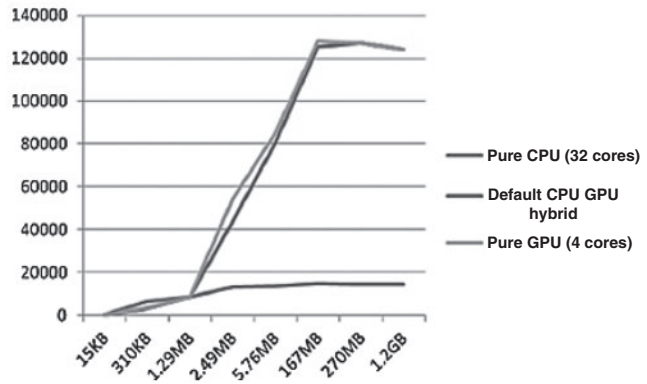

Figure 15: Variation of calculation speed with a file size

The number of PSK entries corresponding to the several dictionary packages used in the experiment are 15 kB (1808 entries), 310 kB (36984 entries), 1.29 MB (163199 entries), 2.49 MB

(271684 entries), 5.76 MB (569790 entries), 167 MB (14867568 entries), 270 MB (21780000 entries) and 1.2 GB (100000 entries).

It can be seen that the dictionary running package that calls GPU resources has a considerable degree of practical significance. Under the same circumstances, performance of a single-core GPU to a single-core CPU can be increased by about 80 times. However, under the hardware conditions of a server with a single dual CPU and four GPUs configuration, it will take about 200 years to achieve an exhaustive combination of 8-digit numbers with uppercase and lowercase letters, which is not time-sensitive and impractical. To achieve minute-level exhaustion, tens of thousands of servers with the same configuration are required, which are expensive and undesired.

It was found that the core occupancy rate of four GPUs is always between 40% and 42%, which is effectively less than 100%. The reason for this may be that Pyrit only implements the use of GPU resources for the parallel part of the Hash algorithm in the handshake packet calculation process. In addition, the calculation method has not allocated the resources of the GPU based on the CUDA architecture to the extent of the best use, which may also be one reason for the low GPU occupancy rate.

6 Conclusions

This article proposed an improved scheme for cracking password in Wi-Fi networks. There are two ways to improve the operating efficiency. One is to use prior knowledge to restrict the combination range of dictionary files. We have verified that it reduces the number of dictionaries set after constraints compared to before constraints. Although it may reduce the hit rate, it can improve the efficiency of cracking within an acceptable range. The other is to use GPU and CPU to perform simultaneous parallel division of task to run dictionary package calculation comparisons to decrypt Wi-Fi's WPA/WPA2 protocol passwords. The speed of dictionary cracking is much faster than the traditional method of relying on CPU computing power. For commonly used weak and short passwords, Wi-Fi passwords can be cracked much faster than a pure CPU. However, this method is obviously helpless for long passwords. Therefore, although the current WPA/WPA2 protocol is more secure, we do not recommend it for users to use short and weak passwords. This article still recommends that users set the password to a high-strength with a long number of digits. In addition, the next work will study the parallel computing resource allocation of multiple servers and discuss the implementation of the PMK computation method for known SSIDs, and then run the dictionary package. Performance improvements and appropriate dictionary combinations will give ideas for how to improve the dictionary, hit rate, etc.

Acknowledgement: The authors would like to thank the editors and reviewers for their valuable suggestions and recommendations.

Funding Statement: This work is supported by SUT research and development fund.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] M. Husak, J. Komarkova, E. Bou-Harb and P. Celeda, "Survey of attack projection, prediction, and forecasting in cybersecurity," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 640–660, 2019.

[2] N. Sun, J. Zhang, P. Rimba, S. Gao, L. Y. Zhang *et al.*, "Data-driven cybersecurity incident prediction: A survey," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 2, pp. 1744–1772, 2019.

[3] A. Burg, A. Chattopadhyay and K. Y. Lam, "Wireless communication and security issues for cyber-physical systems and the internet-of-things," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 38–60, 2018.

[4] Y. Chen, X. Wang, Y. Yang and H. Li, "Location-aware Wi-Fi authentication scheme using a smart contract," *Sensors*, vol. 20, no. 4, pp. 1–22, 2020.

[5] A. Dagelić, T. Perković, B. Vujatović and M. Čagalj, "SSID oracle attack on undisclosed Wi-Fi preferred network lists," *Wireless Communications and Mobile Computing*, vol. 2018, pp. 1–15, 2018.

[6] D. Stiawan, M. Y. Idris, R. F. Malik, S. Nurmaini, N. Alsharif *et al.*, "Investigating brute force attack patterns in IoT network," *Journal of Electrical and Computer Engineering*, vol. 2019, pp. 1–13, 2019.

[7] D. Wang, X. Zhang, J. Ming, T. Chen, C. Wang *et al.*, "Resetting your password is vulnerable: A security study of common SMS-based authentication in IoT device," *Wireless Communications and Mobile Computing*, vol. 7849065, pp. 1–15, 2018.

[8] L. Bosnjak, J. Sres and B. Brumen, "Brute-force and dictionary attack on hashed real-world passwords," in *IEEE 41st Int. Convention on Information and Communication Technology, Electronics and Microelectronics*, Croatia, pp. 1161–1166, 2018.

[9] T. Gautam and A. Jain, "Analysis of brute force attack using TC-dataset," in *IEEE SAI Intelligent Systems Conf.*, UK, pp. 76–81, 2015.

[10] M. Farik and A. S. Ali, "Analysis of default password in routers against a brute-force attack," *International Journal of Scientific and Technology Research*, vol. 4, no. 9, pp. 341–345, 2015.

[11] F. Permatasri and U. Eaganthan, "An implementation of WEP/WPA/WPA2 password cracking using fluxion," *International Journal of Advance Research in Science and Engineering*, vol. 7, no. 2, pp. 641–656, 2018.

[12] M. A. Mohammed, A. F. Degadzor, B. F. Effrim and K. A. Appiah, "Brute force attack detection and prevention on a network using wireshark analysis," *International Journal of Engineering Sciences and Research Technology*, vol. 6, no. 6, pp. 26–37, 2017.

[13] S. K. Kulkarni, "A survey of password attacks, countermeasures and comparative analysis of secure authentication methods," *International Journal of Advanced Research in Computer Science and Management Studies*, vol. 3, no. 11, pp. 319–331, 2015.

[14] M. Obaidat, J. Brown, S. Obeidat and M. Rawashdeh, "A hybrid dynamic encryption scheme for multi-factor verification: A novel paradigm for remote authentication," *Sensors*, vol. 20, no. 15, pp. 1–22, 2020.

[15] M. Wang, F. H. Liao, J. Lin, L. Huan, C. Gu *et al.*, "The making of a sustainable wireless city? Mapping public Wi-Fi access in Shanghai," *Sustainability*, vol. 8, no. 2, pp. 1–15, 2015.

[16] A. H. Adnan, M. Abdirazak, A. S. Sadi, T. Anam, S. Z. Khan *et al.*, "A comparative study of WLAN security protocols: WPA, WPA2," in *Int. Conf. on Advances in Electrical Engineering*, Dhaka, Bangladesh, pp. 165–169, 2015.

[17] C. P. Kohlios and T. Hayajneh, "A comprehensive attack flow model and security analysis for Wi-Fi and WPA3," *Electronics*, vol. 11, no. 11, pp. 1–28, 2018.

[18] S. Nam, S. Jeon, H. Kim and J. Moon, "Recurrent GANs password cracker for IoT password security," *Sensors*, vol. 20, no. 11, pp. 1–19, 2020.

[19] Y. L. Liu, Z. G. Jin and Z. Chen, "Design and implementation of high-speed brute force for WPA/WPA2-PASK," *Computer Engineering*, vol. 37, no. 10, pp. 125–127, 2011.

[20] T. H. Chang, C. M. Chen, H. W. Hsiao and G. H. Lai, "Cracking of WPA and WPA2 using GPUs and rule-based method," *Intelligent Automation and Soft Computing*, vol. 25, no. 1, pp. 183–192, 2019.

[21] H. H. Holm, A. R. Brodtkorb and M. L. Saetra, "GPU computing with Python: Performance, energy efficiency and usability," *Computation*, vol. 8, no. 1, pp. 1–24, 2020.

[22] L. Y. Lei and J. Z. Gang, "Distributed method for cracking WPA/WPA2-PSK on multi-core CPU and GPU architecture," *International Journal of Communication Systems*, vol. 28, no. 4, pp. 723–742, 2013.

[23] Z. Xia, P. Yi, Y. Liu, B. Jiang, W. Wang *et al.*, "GENPass: A multi-source deep learning model for password guessing," *IEEE Transactions on Multimedia*, vol. 22, no. 5, pp. 1323–1332, 2020.