International Journal of Engineering & Technology, 7 (2.5) (2018) 23-26

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research Paper

A novel secret key generation based on image link

A.H. Sulaiman [1]*, I.F.T. Al-Shaikhli[1], M.R. Wahiddin[1], S. Houri[1], N. Jamil[2], A.F. Ismail[1]

[1]Department of Computer Science, Kulliyyah of Information and Communication Technology, International Islamic University Malaysia, 53100 Gombak, Selangor, Malaysia
[2]Department of System and Network, College of Information Technology, Universiti Tenaga Nasional, Jalan Ikram-Uniten, 43000 Kajang, Selangor, Malaysia
[3]Department of Electrical and Computer Engineering, Kulliyyah of Engineering, International Islamic University Malaysia, 53100 Gombak, Selangor, Malaysia
**Corresponding author E-mail: hadisulaiman4@gmail.com*

Abstract

One of the main problems with symmetric encryption is key distribution especially when involving large number of users i.e to generate identical keys at different locations. To address this challenge, we proposed a novel algorithm of secret key infusion protocol (SKIP) to generate an identical secret key. While, the key is generated based on a provided image link, starting pattern and string length which must be kept in secret as the algorithm is publicly known. The image from website must be a static image and used as the input of random bits to produce string of hexadecimal values. In a case where image link is compromised, the adversary has to guess other layers of parameters in starting pattern and string length. The generated secret keys were identical at two different locations. In other observation, different secret keys were generated even with the same image link and pattern length but different starting pattern.

1. Introduction

In a secure communication, a secret key must be known only to the sender and recipient to provide data confidentiality [1]. The same secret key is used in a symmetric encryption system for encryption and decryption. A processing time in symmetric encryption is very fast as compared to asymmetric encryption. However, the symmetric encryption has a problem with key exchange and key distribution, as the key must be highly secured. On the other hand, the issue of secret exchange between the entities involved and the inefficiency of Public Key Cryptography in terms of power and computing resources has initiated a closer look into finding ways of generating symmetric keys without the need to exchange them. Thus, it is necessary to venture a method to resolve the issue of key distribution in symmetric encryption system.

Another feasible way to secure data is by using Quantum Key Distribution (QKD) which can distribute the secret key based on the laws of quantum physics. The QKD was used to produce and distribute keys at highly secure communication within an optical fiber link based on a quantum mechanics. However, the QKD is only limited to optical fiber and hardly realized due to high cost and does not immune to light disturbance. Other disadvantages are impractical and heavily reliant upon monitoring of signal disturbance [2]. Thus, in term of security design, a low cost, feasible and simple protocol has to be realized to establish the same key for encryption and decryption process. Our work deals with those problems in a more flexible and less expensive way.

In 1999, Internet of Thing (IoT) was first introduced by Auto-ID Center and was complimented as a potential strategic newly-emerged corporation [3]. Nowadays, the technology of IoTs has grown tremendously in data transmission that led to heavier traffic within the network of the internet. Currently, there are 9 billion joined devices and it is expected to reach 24 billion devices by 2020 [4]. The challenges of security in IoTs are including limited resources, time-consuming in the encryption process and less network guard [5]. On the other hand, the IoTs offer a great selection of connectivity to capture billions of noise sources as random bits from websites. Thus, there are a huge possibilities of image resources as the random bit for our secret key generation (SKG) due to growth of IoT data. In the meantime time, one of the applications of our secret keys is to highly secure the IoT data.

Many works were carried out in generating the secret key based on different sources of handwriting signature [6], biometric-based handwritten signature [7], electronically steerable parasitic array radiator (ESPAR) antenna [8], physical layer of wireless communication [9] and a random bit from satellite communication [10]. Other work of key generation is using fingerprint, password, smart card [11] and from noisy radio channel measurement [12]. All the methods discussed above not using image to generate key which easily acquired from webpages. From the best of our knowledge, the research work of secret key generation (SKG) based on image was done in [13] by using textured image which is partitioned into regions and textural features. However, the issue of key distribution in symmetric encryption was not discussed. In addition, one or more parameter for a second and third layers of key security from an image was not implemented. In this work, an identical secret key is produced based on a novel algorithm of SKIP at different location, which solved the issue of key distribution in symmetric encryption system between the sender and recipient. Other advantage of our work is three layers of security, as an adversary can't guess the key if the image link is compromised, due to the need of other parameters in starting pattern and string length.

2. Methodology

Before starting the algorithm, the list of Uniform Resource Locators (URLs) of the image, starting pattern and string length are determined to be fed to the Python program which was run on each computer locally. The method used deals with images available from web pages and uses pixels information as a source of the original bit string. The information from the image obtained from

the website is extracted to provide several pixel values as the world wide web gives a huge pool of URLs to be agreed upon by the two parties involved in the communication. The noise extracted from the pixel values is used as input to run the SKIP. The Python script specified that only jpeg image from website can be chosen to be used as input of random bits. The availability of images with millions of pixels enables a large size of string to be used and more probability of starting pattern to be found. In some scenarios, image entropy can be calculated to investigate the randomness of pixel values inside a picture as well as the pattern of their distribution. The security of secret keys is based on three parameters of image URL, the starting pattern and the string length to be extracted. The secret key is generated by using the SKIP algorithm as follows [9]:

1. Insert an image link, a starting pattern and a string length.
2. Look for the starting pattern; say ef2. Continue, if the starting pattern is found. Otherwise, return to step 1.
3. Extract a specific number of bits (pixel values in a particular sequence), say next 4096 bits.
4. Apply Von Neumann corrector on the resulted string.
5. Apply exlusive OR (XOR) corrector on the resulting bit string of step 4.
6. Apply hash functions of MD5, SHA1 and SHA512.
7. Go back to step 1 to generate other key.

More details on the SKIP are as follows. The simulation of SKIP algorithm is based on a static image as the input source. Firstly, an image is downloaded from the image link to a local drive. The image is loaded (offline) line by line for efficiency and memory management. The search of starting pattern is done line by line from middle (one third) of the image until the end of image. The random bits of the first one third of the image is ignored. If the set starting pattern is found, the program will produce random bits in hexadecimal digits. The starting pattern can be anything as long as it is a reasonable hexadecimal values. In the meantime, the string length could be anything from one character to thousands of characters. The resulted bit string is then de-biased by performing correction using Von Neumann corrector as well as XOR corrector. Ultimately, the string is hashed to produce a secret key.

The static image from a web page acts as the source of noise to acquire the random bits. Only a static image is selected as the input because unchanged bits is required, as using dynamic image can vary the bit pattern, thus affecting to different input values. Not all static image from the website can be selected as the input. The best image link have to be the passive website which rarely updates their image, as an active site such as news website sometimes will update and remove the image.

3. Implementation

The simplicity of the method lends itself to a variety of software and hardware implementations. The SKIP provides a non-synchronization technique that ensures the keys generated by all parties involved to be identical, even the SKG is done at different time. To run the SKIP algorithm, we used a programmer of Python which equipped with a request library to communicate with a web server and an image processing library to translate the images. The Python can be run on any operation system (OS) either Windows, Android, Rasbian (OS of Raspberry Pi), etc. We implemented the SKG based on a standalone computer under the operating system of Windows 10. The OS, random access memory (RAM) value and processor speed of the computer to run the Python is Windows 10 (64 bit), 8 GB and 2.6 GHz (Intel i7).

Table 1 shows the types of SKG test at two different locations and internet connectivities. The distance between Semenyih and Gombak is approximately 50 km. The computer in Semenyih was remotely controlled from Gombak using a software of Teamviewer, thus a minimum of one person can perform this key generation for several locations. The key generation was also checked at different wireless networks in wifi, ethernet cable and mobile tethering network in Semenyih for the observation of processing time. In a meanwhile, further verification was also carried out to investigate any disturbance that could occur within the process of communicating to the web page, extracting the image and finding the starting pattern.

Table 1: The test of SKG at different location and type of internet connectivity.

SKG test	Location 1	Location 2
Location	Gombak, Malaysia	Semenyih, Malaysia
Internet connectivity	Ethernet cable	Wifi, ethernet cable and mobile tethering

4. Security of the proposed algorithm

In this algorithm, an adversary; Eve can listen to all communication between Alice and Bob. Eve also knows the algorithm of SKIP. However, the parameters of the protocol are kept secret from her. Due to the enormous pool of possible sources and patterns as well as lengths, we can safely assume that Eve has a minuscule probability of guessing them correctly. In addition, since there is no exchange of keys and key generation occurs locally on each of Alice and Bob's machines, Eve has no way of disrupting the SKIP.

5. Results and discussions

A combination of several websites and starting pattern as well as different string length were used to observe the behavior of the program of SKG based on SKIP implemented in Python. The program used to extract the noise from the image, then started to search the starting pattern. The program also performed filtering process of certain thumbnail and icon images which are usually tiny in size. This is done as a tiny size of image has large probability to found the starting pattern.

The experiment using the algorithm of SKIP was automatically recorded in a Microsoft Excel file. The excel sheet holds the source of URL (path link of the image), the first 32 characters of the found string and the 32 characters of hash value. If no string is found, an "X" is written in the Microsoft Excel file, meaning that no key is generated. The hash value aka the generated key is the main value as well as some timer values for the procedure.

Table 2: The generated identical secret keys based on two locations in Gombak and Semenyih. The starting pattern and string length are the set parameters.

Image	Starting pattern	String length	The first 32 string	Hash value (32 characters)
a	ef2	111	ef2e55689b923906943f0f5c49b6511c	a5cf766846a7dcb4e6a6b5e12d436c23
b	f798a	19	f798a1cb12fbdad9f2f	b9bc4c84c1a6feec3635302a0d200382
C	eeaca	1234	eea-ca5c23e164255820fc5047e642321	5d209a9edda544e14733973da362653d
d	677706	10753	677706bfa27cebe81974e7f9c7baf045	7b3230de35a08cb953ddaf530f8c4104

Table 2 shows the generated keys based on different image link, starting pattern and string length, according to the alphabetic letter of the image links. The data shown in Table 2 was from the Microsoft Excel file which is auto-generated from the Python program. From the table, most hexadecimal digits of starting pattern were found, as the larger the starting pattern, the higher probability of the starting pattern not to be found. The processing time of pattern search, correction and hash were occurred efficiently without any delay. The only delay that could occur was when the program was requesting the image link, starting patern and string

length. The first attempt of starting pattern was full number, for example, 1234. From the table, the string of length was set from minimum of two to five characters, as any value which above one digit can be set. The maximum value of string length is depends on image size. From the experimental results, the generated secret keys were found to be identical even at different locations and sources of internet connectivities. It also shown that the starting patterns can be a combination of numbers, letters and numbers with letters, but limited only to hexadecimal values. The first 32 string is also shown, but if the set string length is 19 (as shown in image b), the display is according to the set value. At the moment, the hash value is set 32 characters. In the future, the character can be expanded or reduced according the need of key length for encryption and/or decryption process.

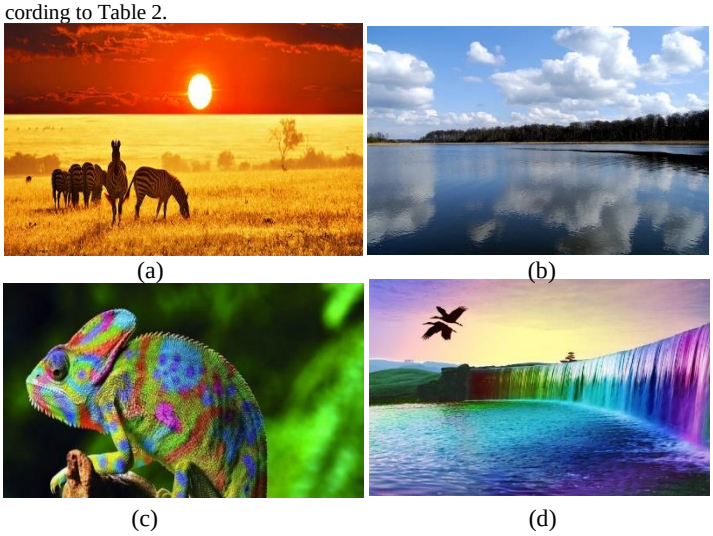
Table 3: The generated key at variation of starting pattern based on the same image link of https://jpeg.org/images/jpegls-home.jpg and string length of 111

Starting pattern	The first 32 string	Hash value (32 characters)	Time of VNC (ms)	Time of XOR (ms)
2c	2c20773d235e2 4483175493080 90a181	b5ae2a90aeea b9d68deede0 9454e537a	0.0635	0.0632
bee	bee2877a61040 d94506d127882 bd3902	70b5464b0cd 23d70dc01d2 e7d15aa32a	0.0805	0.0930
e07e	e07ec93f4e1b62 42f1313d2334f 9e2c7	60bb3973123 1dbc901b43a 9f9ac8154b	0.0872	0.0510
6eeac	6eeac5208d4b0 2fcd0cb183f172 51bb2	bc22d0e33e0 e83017573f8 441b6d9130	0.0802	0.0661
06d12	06d127882bd39 027e914fe6c89 20ab25	d28d9d99b0c 0d6ea50c56fd 2b51ee4d4	0.1113	0.0933

Table 3 shows the hash value and the first 32 string of hexadecimal value based on the same image link (https://jpeg.org/images/jpegls-home.jpg) and the same starting pattern of 111 but different starting pattern. The time to do Von Neumann corrector and XOR are also displayed. Any kinds of disturbance during the image acquirement were also observed as it might occured while loading the image from a different location and while extracting information from it. From our experiment, less than five seconds was taken in generating the secret key. The secret keys are not purposely to share and can be easily regenerated whenever the key is compromised.

Fig. 1(a-d) show the image samples in regards to Table 2. Note that, this static image will not always available as it might have been removed, has its name changed or is temporarily unavailable. Whenever the image link is not available, the other link will be acted as a backup. The static image can be any size, as the important is the parameter setting. However, we need to ensure that the program do not detect any thumbnail or tiny size of image because with lesser string length, it is more difficult to find the set pattern. Bigger jpeg image has longer string of hexadecimal value, thus easier to generate the secret key. The unavailability of image link isn't really matter, as billions of JPEG images are available from the website. In our proposed plan, the program will be developed for Android application so as to easily generate secret key form mobile phone.

Fig. 1: The static images based on provided image link (a,b,c and d) according to Table 2.



(a) (b)

(c) (d)

6. Conclusion

We have successfully generated the identical secret key based on two locations, assisted by a simple and practical algorithm of SKIP. This novel algorithm solved the issue of key distribution in symmetric encryption system. The adversary can't guess the key even the algorithm is known, since the key requires the image link and the parameters. Even though the image link is compromised, the second and the security layer of starting pattern and string length,respectively must be known, otherwise the key is unknown. Whenever any sides have the same image link and the parameters, the secret keys are found identical even at a different location. More than two locations can generate the identical secret key without the need to transport the key. The identical secret keys were also generated at the same link and string length even with different starting pattern.

Acknowledgements

This work was partially supported by International Islamic University Malaysia, FRGS14-127-0368 and ERGS13-018-0051 from Ministry of Higher Education of Malaysia.

References

[1] Ren, K., Su, H., and Wang, Q. 2011. Secret key generation exploiting channel characteristics in wireless communications. IEEE Wirel. Commun., 18(4), 6–12. DOI: 10.1109/MWC.2011.5999759;

[2] Takesue, H., Sasaki, T., Tamaki, K., and Koashi, M. 2015. Experimental quantum key distribution without monitoring signal disturbance. Nat. Photonics, 9(12), 827–831. DOI: 10.1038/nphoton.2015.173;

[3] Ning, H., and Wang, Z. 2011. Future Internet of Things architecture: like mankind neural system or social organization framework? IEEE Commun. Lett., 15(4), 461–463. DOI: 10.1109/lcomm.2011.022411.110120;

[4] Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. 2013. Internet of Things (IoT): a vision, architectural elements, and future directions. Futur. Gener. Comput. Syst., 29(7), 1645–1660. DOI: 10.1016/j.future.2013.01.010;

[5] Jing, Q., Vasilakos, A. V., Wan, J., Lu, J., and Qiu, D. 2014. Security of the Internet of Things: perspectives and challenges. Wirel. Networks, 20(8), 2481–2501. DOI: 10.1007/s11276-014-0761-7;

[6] Feng, H., and Wah, C. C. 2002. Private key generation from on-line handwritten signatures. Inf. Manag. Comput. Secur., 10(4), 159–164. DOI: 10.1108/09685220210436949;

[7] Freire-Santos, M., Fierrez-Aguilar, J., and Ortega-Garcia, J. 2006. Cryptographic key generation using handwritten signature. Proceeding SPIE 6202, Biometric Technol. Hum. Identif. III, 62020N–1–62020N–7. DOI: 10.1117/12.665875;

[8] Aono, T., Higuchi, K., Ohira, T., Komiyama, B., and Sasaoka, H. 2005. Wireless secret key generation exploiting reactance-domain scalar response of multipath fading channels. IEEE Trans. Antennas Propag., 53(11), 3776–3784. DOI: 10.1109/TAP.2005.858853;
[9] El Hajj Shehadeh, Y., and Hogrefe, D. 2014. A survey on secret key generation mechanisms on the physical layer in wireless networks. Secur. Commun. Networks, 8(2), 332–341. DOI: 10.1002/sec;
[10] Wahiddin, M. R., Shanaz Noor Sham, N. S., Saeb, M., and Mior Hamdan, M. H. 2010. A protocol for secret key infusion from satellite transmissions. Int. J. Comput. Netw. Secur., 2(7), 99–102.
[11] Ahmed, F. and Siyal, M. Y. 2005. A novel approach for regenerating a private key using password, fingerprint and smart card. Inf. Manag. Comput. Secur., 13(1), 39–54. DOI: 10.1108/09685220510582665;
[12] Patwari, N., Croft, J., Jana, S., and Kasera, S. K. 2010. High-rate uncorrelated bit extraction for shared secret key generation from channel measurements. IEEE Trans. Mob. Comput., 9(1), 17–30. DOI: 10.1109/TMC.2009.88;
[13] Omotosho, A., and Emuoyibofarhe, J. 2015. Private key management scheme using image features. J. Appl. Secur. Res., 10(4), 543–557. DOI: 10.1080/19361610.2015.1069642;