AAVSS: Auxiliary Authorization Mechanism Based on Arbitrary-Angle VSS

Ying Chin Chen[1], Kuo Jui Wei[1], Jung San Lee[1], Ngoc Tu Huynh[2] and Jyun Hong Lin[1]
(Corresponding author: Jung San Lee)
Department of Information Engineering and Computer Science, Feng Chia University[1]
Taichung, 40724, Taiwan
Faculty of Information Technology, Ton Duc Thang University[2]
Ho Chi Minh City, Vietnam
(Email: leejs@fcu.edu.tw)
(Received Nov. 28, 2017; Revised and Accepted May 5, 2018; First Online Mar. 2, 2019)

Abstract

To provide seamless Internet services, most public buildings, including coffee shops, airports, and libraries, temporary personal computers are offered to users for network access. This has led to a potential risk that the secret information of the user may be leaked out once these temporary computers have been affected by Trojans. Generally, the service provider checks the authority of a user according to a series of authentication procedure. While a user enters the verification token into the public computers, an attacker may apply a key-logger to steal the password or personal information. In this article, we have first introduced the visual secret sharing technique with arbitrary-angle stacking to design an auxiliary authorization protocol. According to the stacked one-time password, users can have the access to network services without keying any secret information into the public computers. Moreover, the efficiency of AAVSS is favorable to resource-constrained mobile device.

Keywords: Auxiliary Authorization; Keylogger; Nearest-Neighbor Interpolation; VSS

1 Introduction

As the Internet brings convenience for the whole world, human beings now can purchase clothes online, chat online, organize conference online, watch the pay-TV, and enjoy other online products expediently. Corresponding to this trend and due to the heavy work burden on people in the modern society, it is in great need that people can access to various services or products just through the Internet without having to go out personally.

In recent decades, various kinds of network services are offered via the Internet. People expect that they are able to obtain the services everywhere and anytime. Thus, the verification mechanisms are necessary for confirming legibility of both service provider and user. Among those mechanisms, the password authentication [2–6, 9, 11–13] is an easy way to achieve this purpose. In such environment, a new user has to provide a pair of identity and password to the service provider in the registration. The server then keeps the secret information in the database once it has accepted the joining request. After that, the server maintains the service access according to the comparison between the received authentication token and the recorded one in the database.

Nevertheless, researchers have pointed out that this simple mechanism might suffer from the stolen verifier attack. Besides, it requires a large amount of memory to maintain the password table and corresponding information. Hereafter, the smart card has been introduced in the design of authentication mechanisms to solve this security problem and mitigate the storage consumption. Personal information and authentication token are kept in the card instead of the server database for validity proof. Hence, the risk of stolen verifier attack could be eliminated.

Subsequently, there have many attacks based on the information retrieved from the smart card. According to the extracted token, intruders can further mount malicious attacks, such as forgery attack and impersonation attack. In addition, two-factor authentication schemes have been designed for enhancing the entropy of verification token, in which the difficulty in compromising the system could be highly reinforced. Aside from a smart card, people adopt the fingerprint as the other factor for personal information protection. The sampling process of fingerprint, however, is a tough problem in the implementation. It is due to the high sensitivity of cryptographic function.

Actually, the validity is not the only essential requirement that these authentication mechanisms have to achieve. Many new challenges including anonymity, untraceability, efficiency, and resistance to recent attacks, have come out in designing a novel authentication mech-

Table 1: VSS stacking

s_1	s_2	$s_1 \vee s_2$
■	□	■
■	■	■
□	■	■
□	□	□

anism. Among them, a common situation is seldom considered in the field of authentication mechanism. That is, people often need to login a network system via a public computer or some other persons laptops. In this case, it is hard to guarantee that the login information could be well-protected. An attacker can install a key-logger program into a public computer; Thus, recording the identity and password of a login user [7, 11]. Once people cannot get rid of this temporary switch situation, how to prevent a personal secret from being intercepted or recorded must be firmly concerned in developing an authentication system.

In this article, we aim to propose an auxiliary authority mechanism based on the visual secret sharing (VSS) technique [8, 10], in which a user can use the mobile phone to switch a service to a temporary computer instead of entering any personal secret. For simplicity, the abbreviation of the new mechanism is defined as AAVSS. More precisely, a user can login the network systems with personal computer by a pre-defined authentication scheme. Whenever the personal computer is inaccessible, the user can employ the smartphone to obtain the one-time token (OTP) to have the access [3]. In AAVSS, the nearest-neighbor interpolation (NNI) is integrated into the VSS to achieve the arbitrary-angle stacking, while a smartphone is applied to record the base of VSS [1, 15]. According to International Telecommunication Union (ITU) statistics in 2015, there are more than 7.085 billion smartphone subscribers around the world [14]. The adoption of smart phone does make sense while being integrated into an authentication mechanism. Moreover, the computing ability of a smart phone is much higher than that of a smart card. This device can share parts of computation for verification. In particular, the property of arbitrary-angle stacking of VSS is difficult to complete; thus, it is the main challenge in developing AAVSS.

The rest of this article is organized as follows. Related works are introduced in Section 2, followed by the details of AAVSS in Section 3. The performance analysis is explained in Section 4. We make conclusions in Section 5.

2 Related Works

In the following, we introduce the concept of VSS and NNI. The VSS is used to embed OTP content into shares. Hereafter, a legal user can figure out OTP by stacking shares on the base kept in the smartphone. As to the NNI, it is adopted to achieve the arbitrary-angle stacking.

Table 2: The stacking principle of Lin *et al.*'s scheme

GS_1	GS_2^0	GS_2^{120}	GS_2^{240}	SP_1	SP_2	SP_3
□	□	□	□ ■	□	□	□ ■
		■	□ ■		■	□ ■
	■	□	□ ■	■	□	□ ■
		■	□ ■		■	□ ■
■	□	□	□ ■	■	■	■ □
		■	□ ■		□	■ □
	■	□	□ ■	□	■	■ □
		■	□ ■		□	■ □

2.1 Visual Secret Sharing

The VSS technique is first proposed by Naor and Shamir [10]. It distributes the pixel of a secret image SP to two transparent shares TS_1 and TS_2. S_1 and S_2 denote the corresponding position pixels of TS_1 and TS_2. The stacking operation is the OR boolean operation ($\vee$), as shown in Table 1. People cannot learn anything useful information of SP from only one share. By stacking those two transparencies, the content of SP can be revealed.

In 2014, Lin *et al.* have extended this idea to develop a multi-secret VSS mechanism based on random grid [8], in which the secrets can be shown by specific angle stacking. Given three secret images SP_1, SP_2, and SP_3, they randomly generate a grid base GS_1 and three temporary shares GS_2^0, GS_2^{120}, GS_2^{240} according to the principle in Table 2. Note that Lin *et al.* apply the exclusive-or operation to stack shares instead of OR function. How to construct those shares depends on two rules:

1) If the pixel of secret image is white, the pixel at the corresponding position of share must be the same as that of GS_1.

2) If the pixel of secret image is black, the pixel at the corresponding position of share must be the opposite one of GS_1.

Hereafter, a user who collects base GS_1 and share GS_2 is able to reveal SP1. Rotating GS2 by 120°, the user further extracts SP_2. Finally, the user can figure out SP_3 by rotating GS_2 by 240°.

2.2 Nearest-Neighbor Interpolation

The NNI is a technique used to adjust the scale of image or to rotate an image [1, 15]. In AAVSS, we apply it to re-define the pixel coordinates of share after rotating. Suppose that the new pixel coordinate is (x', y'), as illustrated in Figure 1. Due to the rotation operation, the

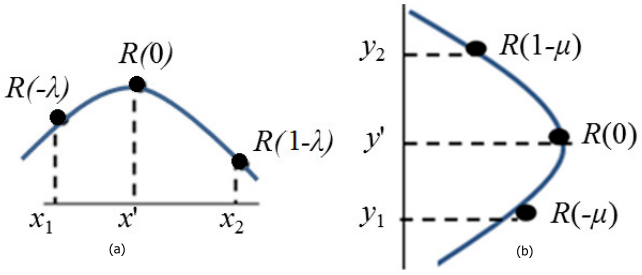


Figure 1: The new coordinate (x', y') of pixel after rotating

values of (x', y') might not be integers. Note that (x_1, y_1) and (x_2, y_2) are the neighbor coordinates with integer appearance. Thus, we can use NNI to transfer these values into integers for fulfilling the image format. We first calculate the difference between (x', y') and neighbors based on Equation (1). Subsequently, the new coordinate is modified as (x", y") according to Equation (2) and Equation (3). The function R(u) is used to check the nearest coordinate of (x', y'), and u is the parameter for distance judgment.

$$\left\{ \begin{array}{l} \lambda = x' - x_1, x_1 \le x' \le x_2 \\ \mu = y' - y_1, y_1 \le y' \le y_2 \end{array} \right\} \quad (1)$$

$$\left\{ \begin{array}{l} x'' = R(-\lambda)x_1 + R(1-\lambda)x_2 \\ y'' = R(-\mu)y_1 + R(1-\mu)y_2 \end{array} \right\} \quad (2)$$

$$R(u) \quad = \quad \left\{ \begin{array}{l} 1 \text{ if } -0.5 \le u \le 0.5 \\ 0 \text{ otherwise} \end{array} \right\} \quad (3)$$

Figure 2 is an example illustrating how to decide the new coordinate. Black dots of Figure 2(a) are the original coordinates, while white ones are the rotated positions. Suppose $(x', y') = (2.7, 1.4)$ in Figure 2(b). (x_1, y_1), (x_1, y_2), (x_2, y_1) and (x_2, y_2) are (2, 1), (2, 2), (3, 1), and (3, 2), respectively. Thus, we can have $(x'', y'') = (3, 1)$ according to the above equations.

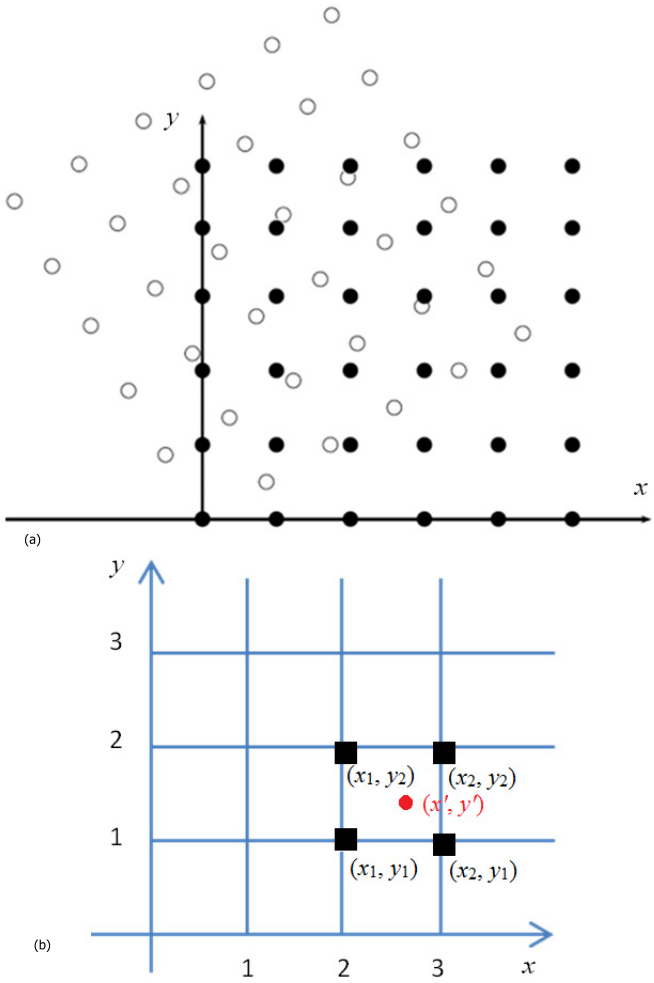


Figure 2: Example of coordinate re-definition

3 The Details of AAVSS

In this section, we describe the details of the auxiliary authority mechanism, including the setup phase, transference phase, and VSS share generation phase. In the first phase, we explain the environment setup and essential assumptions. The authority transference protocol is introduced in the second phase. How to achieve the arbitrary-angle stacking is presented in the final phase.

3.1 Setup Phase

There are four roles in AAVSS, MU (Mobile User), AS (App Server), PC (Public personal Computer), and S (Service provider). MU is a mobile subscriber with an intelligent cellular phone containing a secret base S_1. S_1 is shared between AS and MU. AS is an App server which is responsible for generating VSS base and share. MU has to register at AS and installs an App connecting to AS. S is a server offering a specific service over the Internet. AS and S have shared a secure channel, while MU is a registered user of S and possesses a pair of identity and password. PC is the so called non-personal computer. Notations used in the article are defined in Table 3.

3.2 Transference Phase

Once MU switches to a PC, MU can obtain an OTP token to access network service instead of entering a real password according to the following procedure. The flowchart of the transference phase is illustrated in Figure 3.

Step 1. MU starts the App and sends a request to AS, including ID_{MU}, IDs, and RS.

Step 2. AS generates a four-digit secret number FS and sends it to S along with ID_{MU} via a pre-shared secure channel. Note that FS is an OTP for this pro-

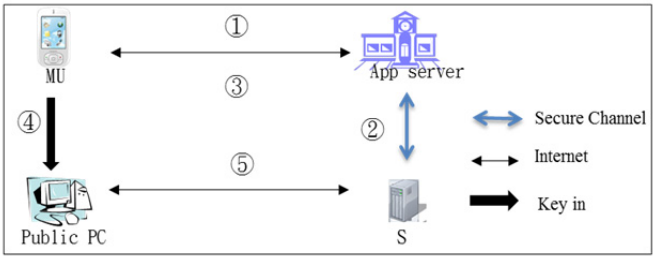


Figure 3: Transference flowchart

Table 3: Notations used in AAVSS

IDs	The identity of service provider S
ID_MU/PW_MU	The identity/password of MU for S
RS	A random seed for generating four rotation angles(R_1, R_2, R_3,R_4)
FS	A four-digit secret number, $FS = P_1P_2P_3P_4$
S_1	The base of MU

tocol run. AS then figures out four rotation angles (R_1, R_2, R_3, R_4) according to the received RS. With these four angles, AS employs the procedure of subsection 3.3.2 to construct four temporary shares S_{2a}, S_{2b}, S_{2c}, and S_{2d} containing P_1, P_2, P_3, and P_4, respectively.

Step 3. AS returns $S_2 = S_{2d}$ to MU.

Step 4. MU applies RS to obtain (R_1, R_2, R_3, R_4) and then stacks S_2 on S_1. Rotating S_2 with R_4, P_4 could be generated. Keeping on rotating R_3, R_2, and R_1 sequentially, MU could have P_3, P_2, and P_1. MU keys in ID_{MU} and FS= (P_1, P_2, P_3, P_4) into the web page on PC.

Step 5. S checks if (ID_{MU}, FS) is the same as the one sent from AS. If they are correct, S accepts the request; otherwise, the connection is terminated.

3.3 VSS Share Generation Phase

Here we explain how to achieve the arbitrary-angle VSS stacking. As we use the concept of strongbox to lay out the OTP content, all the shares and base are in the shape of circle. The secret base S_1 kept in MU and AS is constructed in subsection 3.3.1, while the share S_2 is generated in subsection 3.3.2. All pixels are divided into eight types of 3×3 block, as shown in Table 4. Note that P_1, P_2, P_3, and P_4 are displayed in four pictures, and the content of these four pictures are located separately. If the stacked result of a block contains four white pixels, it is regarded as the white pixel of an OTP digit. In case that the stacked results of a block possesses nine black pixels, it is considered as the black pixel of an OTP digit.

Table 4: Block types

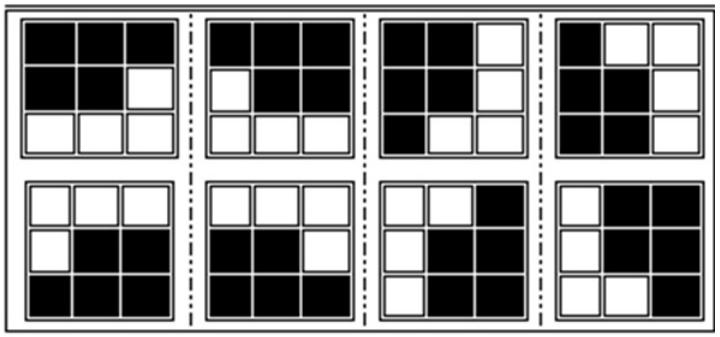
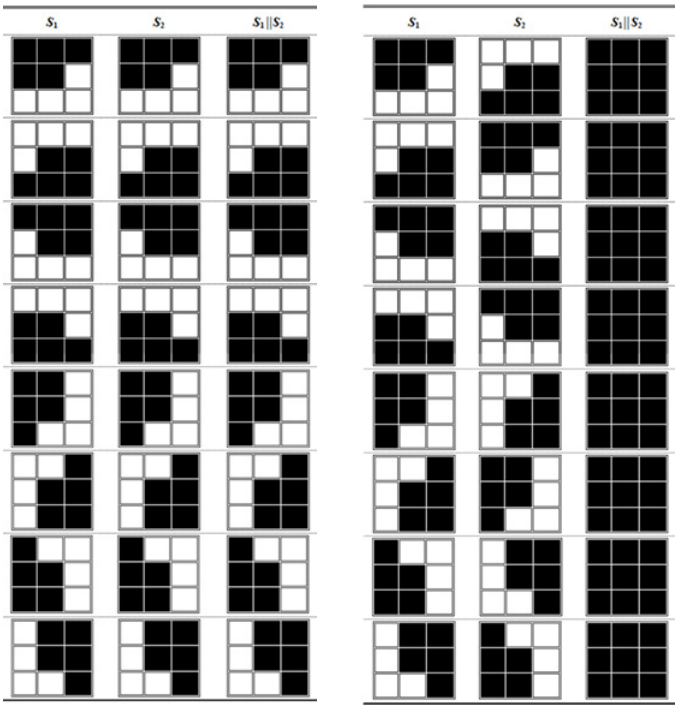


Table 5: Stacking rules

(a) Stacking rules of white blocks (b) Stacking rules of black blocks



3.3.1 The Generation of S_1

Once AS received and accepted the registration of MU, it randomly selects blocks from Table 4 to generate a secret base S_1, as displayed in Table 6(a). S_1 is sent to MU and is kept in its database related to ID_{MU}.

3.3.2 The Generation of S_2

The S_2 consists the blocks from Table 4 according to the rules of Table 5. For a white pixel of an OTP digit, AAVSS generates a block of share corresponding to the position of S_1 based on Table 5(a) such that the stacked result ($S_1 \vee S_2$) could be white. As to a black pixel of an OTP digit, AAVSS picks a block of share corresponding to the position of S_1 based on Table 5(b) such that the stacked result ($S_1 \vee S_2$) could be black.

In the following, we describe how to generate four tem-

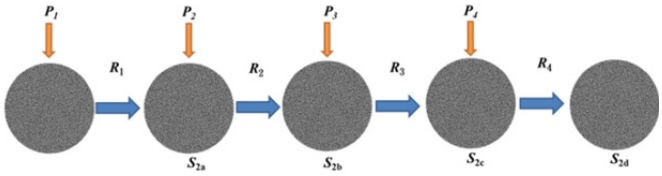


Figure 4: The procedure of generating four shares

porary shares to achieve arbitrary-angle stacking. The procedure is illustrated in Figure 4.

Step 1. $P_1P_2P_3P_4$. AAVSS random picks up four digits(P_1, P_2, P_3, P_4) as the OTP of this section.

Step 2. S_{2a}. AAVSS refers to the block positions of S_1 and P_1 to generate corresponding block of the first share. For a white pixel of P_1, AAVSS randomly selects a block of Table 4 for the corresponding position in the share. As to a black pixel of P_2, it picks the opposite block of S_1 for the corresponding position in the share. According to NNI, it rotates this share by the angle $(-R_1)$ to form S_{2a}.

Step 3. S_{2b}. AAVSS refers to the block positions of S_1 and P_2 to generate corresponding block of the second share. Note that we only consider the black pixel of P_2, P_3, and P_4 to avoid pixel interference problem. For a black pixel of P_2, AAVSS selects the opposite block of S_1 and replaces the corresponding block of S_{2a} with this one. According to NNI, it rotates this share by the angle $(-R2)$ to form S_{2b}.

Step 4. S_{2c}. AAVSS refers to the block positions of S_1 and P_3 to generate corresponding blocks of the third share. For a black pixel of P_3, AAVSS chooses the opposite block of S_1 and replaces the corresponding block of S_{2b} with this one. According to NNI, it rotates this share by the angle $(-R_3)$ to form S_{2c}.

Step 5. S_{2d}. AAVSS refers to the block positions of S_1 and P_4 to generate corresponding block of the last share. For a black pixel of P_4, AAVSS finds the opposite block of S_1 and replaces the corresponding block of S_{2c} with this one. According to NNI, it rotates this share by the angle $(-R_4)$ to form S_{2d}.

Step 6. $S_2 = S_{2d}$, as shown in Table 6(b).

3.4 Performance Analysis

To prove the practicability of AAVSS, we have simulated the system to examine the performance. In particular, the sizes of base and share are set to be 128×128 pixel, 192×192 pixel, 256×256 pixel, 384×384 pixel, and 512×512 pixel, which are suitable for the mobile device. We use desktop to simulate the AS. CPU used for the server is AMD FX-6300 Six-Core 3.5GHz with 8GB RAM.

Table 6: S_1 and S_2

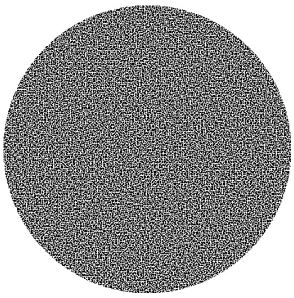
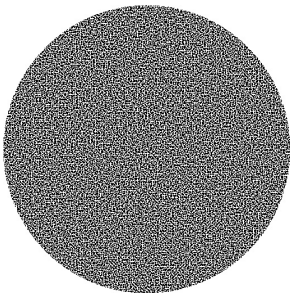
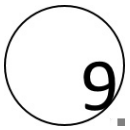
(a) S_1	(b) S_2
	

Table 7: Secret digits of an OTP

P_1	P_2	P_3	P_4
			

The operating system is Window 7 with 64bit. The used program language is Microsoft Visual Studio 2013 C++. The examination includes the recognition of stacked results and the generation efficiency of base and share. We first generated ten OTP's (i.e., FS's) which contain four digits (P_1, P_2, P_3, P_4). Each four-digit number is spread on four pictures, as shown in Table 7.

For each base size, we have generated a base S_1 according to Table 4. The results are displayed in Table 8. For each base, we have generated two sets of (S_{2a}, S_{2b}, S_{2c}, S_{2d}) according to two sets of (P_1, P_2, P_3, P_4) and two sets of rotation angle (R_1, R_2, R_3, R_4). The outcomes are illustrated in Table 9. Undoubtedly, nothing could be revealed. The details of rotation angles and stacked outcomes are listed in Table 10. Here is an example to illustrate how to reveal the OTP in the first row of Table 10. Given the set of rotation angle (139°, 192°, 85°, 301°), we first stacked S_2 on S_1 and rotated S_2 by 301°. Then we obtained a recognizable digit 4. We kept on rotating S2 by 85° to have digit 9, by 192° to reveal digit 2, and by 139° to extract digit 3. Note that 3294 is the OTP for this protocol run. It is clear that all the operations could lay out a recognizable digit from the human vision perception. Thus, the arbitrary-angle stacking is confirmed in AAVSS.

To highlight the contribution of AAVSS, we also inspected the efficiency for constructing base and shares.

Table 8: S_1 with different size (Pixel)

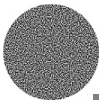
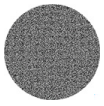
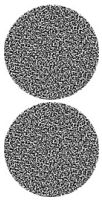
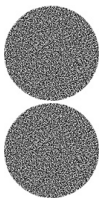
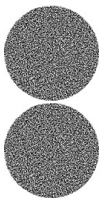
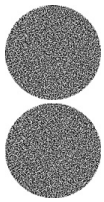
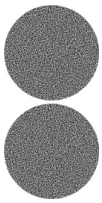
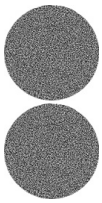
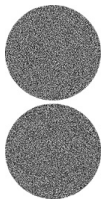
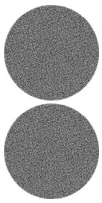
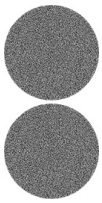
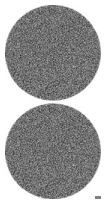
128×128	192×192	256×256	384×384	512×512
				

Table 9: S_2 with different size (Pixels)

Size	S_{2a}	S_{2b}	S_{2c}	S_{2d}
128 × 128				
192 × 192				
256 × 256				
384 × 384				
512 × 512				

For each size of test image, we performed the system two thousand times to gather the statistics. Two thousands random seeds are generated to produce two thousand sets of rotation angles. Similarly, two thousand sets of secret numbers (OTPs) are embedded into corresponding shares. The average time each step is shown in Table 11.

The generation of base S_1 is quick as it is constructed randomly according to the blocks of Table 4. The time for producing the first temporary share S_{2a} is longer since we need to consider the black and white pixels of the secret digits. By contrast, the time for creating S_{2b}, S_{2c}, and S_{2d} is shorter as only the black pixels of secret digit are referred to.

In Table 11, S_2 displays the time for completing a share that will be transferred over the Internet. Namely, it is the time summation of generating those four temporary shares. We have selected five sizes of test image, which are suitable for smartphone appearance. Actually, it is easy for people to recognize the content of stacked result,

Table 10: Stacked results

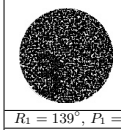
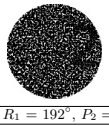
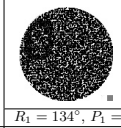
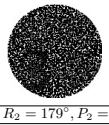
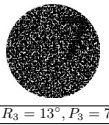
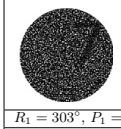
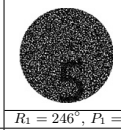
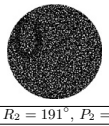
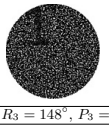
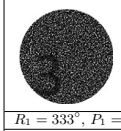
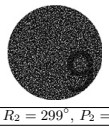
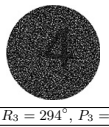
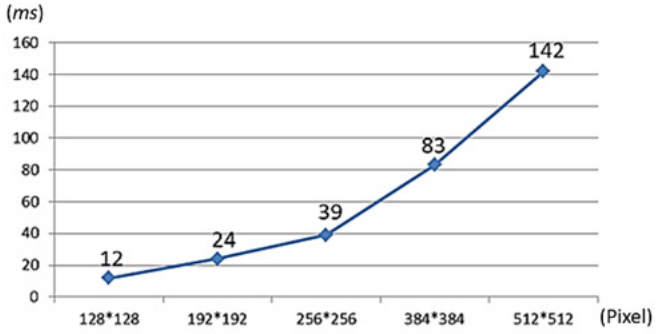
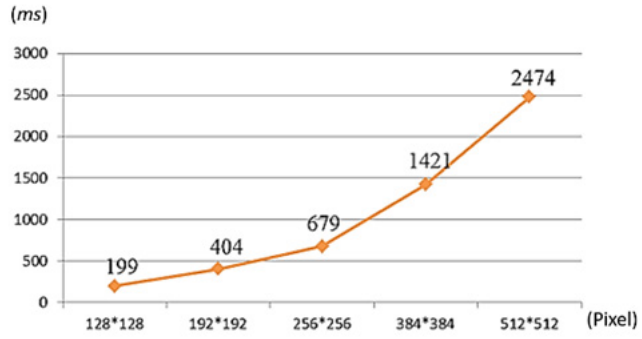
Size	$S_1 \vee S_{2a}$	$S_1 \vee S_{2b}$	$S_1 \vee S_{2c}$	$S_1 \vee S_{2d}$
128 × 128	 $R_1 = 139°, P_1 = 3$	 $R_1 = 192°, P_2 = 2$	 $R_3 = 85°, P_3 = 9$	 $R_4 = 301°, P_4 = 4$
	 $R_1 = 134°, P_1 = 0$	 $R_2 = 179°, P_2 = 3$	 $R_3 = 13°, P_3 = 7$	 $R_4 = 131°, P_4 = 9$
192 × 192	 $R_1 = 303°, P_1 = 7$	 $R_2 = 114°, P_2 = 6$	 $R_3 = 173°, P_3 = 4$	 $R_4 = 221°, P_4 = 9$
	 $R_1 = 246°, P_1 = 5$	 $R_2 = 191°, P_2 = 0$	 $R_3 = 148°, P_3 = 1$	 $R_4 = 65°, P_4 = 8$
256 × 256	 $R_1 = 333°, P_1 = 3$	 $R_2 = 299°, P_2 = 9$	 $R_3 = 294°, P_3 = 4$	 $R_4 = 346°, P_4 = 5$
	$R_1 = 282°, P_1 = 2$	$R_2 = 321°, P_2 = 1$	$R_3 = 187°, P_3 = 7$	$R_4 = 126°, P_4 = 0$
384 × 384	$R_1 = 156°, P_1 = 3$	$R_2 = 194°, P_2 = 5$	$R_3 = 90°, P_3 = 2$	$R_4 = 257°, P_4 = 8$
	$R_1 = 34°, P_1 = 1$	$R_2 = 96°, P_2 = 4$	$R_3 = 25°, P_3 = 9$	$R_4 = 207°, P_4 = 7$
512 × 512	$R_1 = 316°, P_1 = 5$	$R_2 = 279°, P_2 = 8$	$R_3 = 33°, P_3 = 7$	$R_4 = 114°, P_4 = 2$
	$R_1 = 40°, P_1 = 1$	$R_2 = 54°, P_2 = 2$	$R_3 = 254°, P_3 = 9$	$R_4 = 46°, P_4 = 6$

Table 11: Time for generating base and shares

Time *(ms)*						
Size (Pixel)	S_1	S_{2a}	S_{2b}	S_{2c}	S_{2d}	S_2
128×128	12	51	50	49	49	199
192×192	24	107	99	99	99	404
256×256	39	182	166	165	166	679
384×384	83	388	344	345	344	1421
512×512	142	673	600	601	600	2474



(a) S_1



(b) S_2

Figure 5: The fold-line graphs of base and shares

as shown in Table 10, no matter what kinds of size we selected. Taking the size of 256×256 for instance, the average time for creating a share for MU is only 679 ms, which is acceptable in integrating a mobile phone as the second factor of authentication. Again, we have provided the fold-line graphs of efficiency under different sizes in Figure 5. They can help to make sense of the practicability of AAVSS in playing an auxiliary authority procedure.

3.5 Conclusions

In this article, we first used VSS technique with arbitrary-angle to develop an auxiliary procedure for preventing real password from being recorded in a public computer. In particular, NNI is adopted to achieve the arbitrary-angle stacking, which can be considered as a strongbox. The security of the applied VSS can be referred to that of [8]. With limited time and trial constraints, AAVSS can effectively produce an OTP to protect secret information when the personal computer is inaccessible.

References

[1] H. C. Chao and T. Y. Fan, "Random-grid based progressive visual secret sharing scheme with adaptive priority," *Digital Signal Processing*, vol. 68, pp. 69–80, 2017.

[2] D. Chen, N. Zhang, Z. Qin, X. Mao, Z. Qin, X. Shen, and X. Y. Li, "S2m: A lightweight acoustic fingerprints-based wireless device authentication protocol," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 88–100, 2017.

[3] N. Haller, *The S/Key One-Time Password System*, RFC 1760, 1995.

[4] A. E. Jr, W. Kantrowitz, and E. Weiss, "A user authentication scheme not requiring secrecy in the computer," *Communications of the ACM*, vol. 17, no. 8, pp. 437–442, 1974.

[5] M. A. Khan, "A survey of security issues for cloud computing," *Journal of Network and Computer Applications*, vol. 71, pp. 11–29, 2016.

[6] D. Kreutz, O. Malichevskyy, E. Feitosa, H. Cunha, R. da R. Righi, and D. D. J. de Macedo, "A cyber-resilient architecture for critical security services," *Journal of Network and Computer Applications*, vol. 63, pp. 173–189, 2016.

[7] T. Limbasiya, M. Soni, and S. K. Mishra, "Advanced formal authentication protocol using smart cards for network applicants," *Computers & Electrical Engineering*, vol. 66, pp. 50-63, 2018.

[8] K. S. Lin, C. H. Lin, and T. H. Chen, "Distortionless visual multi-secret sharing based on random grid," *Information Sciences*, vol. 288, pp. 330–346, 2014.

[9] Y. J. Liu, C. C. Chang, and S. C. Chang, "An efficient and secure smart card based password authentication scheme," *International Journal of Network Security*, vol. 19, no. 1, pp. 1-10, 2016.

[10] M. Naor and A. Shamir, "Visual cryptography," in *Workshop on the Theory and Application of of Cryptographic Techniques*, pp. 1–12, 1994.

[11] S. Sagiroglu and G. Canbek, "Keyloggers," *IEEE Technology and Society Magazine*, vol. 28, no. 3, 2009.

[12] R. Song, "Advanced smart card based password authentication protocol," *Computer Standards & Interfaces*, vol. 32, no. 5-6, pp. 321–325, 2010.

[13] H. M. Sun, Y. H. Chen, and Y. H. Lin, "oPass: A user authentication protocol resistant to password stealing and password reuse attacks," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 651–663, 2012.

[14] S. Teltscher, E. Magpantay, I. Vallejo, L. Kreuzenbeck, D. Korka, V. Gray, D. Olaya, M. Hilbert, M. Minges, N. Delmas, *et al.*, "Measuring the information society," *Geneva: International Telecoms Union*, 2013. (`https://www.itu.int/en/ITU-d/Statistics/Documents/publications/mis2013/MIS2013_without_Annex_4.pdf`)

[15] C. C. Wu, S. J. Kao, and M. S. Hwang, "A high quality image sharing with steganography and adaptive authentication scheme," *Journal of Systems and Software*, vol. 84, no. 12, pp. 2196–2207, 2011.

Biography

Ying-Chin Chen is pursuiting her MS degree in Information Engineering and Computer Science in Feng Chia University, Taichung, Taiwan. Her current research interests include information security and visual secret sharing.

Kuo-Jui Wei received the Ph.D. degree in Information Engineering and Computer Science in 2016 from Feng Chia University, Taichung, Taiwan. His current research interests include information security and mobile communications.

Jung-San Lee has worked as a professor in the Department of Information Engineering and Computer Science at Feng Chia University, Taiwan, since 2017. His current research interests include information security and mobile communications.

Ngoc-Tu Huynh is a Lecturer with Faculty of Information Technology, Ton Duc Thang University, Ho Chi Minh City, Vietnam. Her current research interests include information security and cryptography.

Jyun-Hong Lin received MS degree in information engineering and computer science in Feng Chia University, Taichung, Taiwan in 2015. His current research interests include image processing, and network security.