An Authorization Transfer Protocol for Confidentiality Preserving in Public Access Devices

Yun-Yi Fan[1], Chung-Wei Kuo[1], Tzu-Hao Chen[1], Chia-Hung Chang[1], Jung-San Lee[1,2]*

[1] Department of Information Engineering and Computer Science, Feng Chia University, Taiwan
[2] Cybersecurity Technology Institute, Institute for Information Industry, Taiwan
p1171237@o365.fcu.edu.tw, cwkuo@fcu.edu.tw, wilsonchen.tzuhao@gmail.com,
a19990101152@gmail.com, leejs@fcu.edu.tw

Abstract

The Internet has brought marvelous convenience to people in recent years. In most public buildings, coffee shops, and airports, temporary personal computers are provided to users for network access. This has led to a potential risk that the secret information of the user may be stolen once these temporary computers have been compromised. Generally, a service provider verifies the authority of a user through a verification procedure. Of course, a user has to enter an authentication token into the public computer. Thus, an attacker can apply a key-logger to steal the password or personal information. After that, this attacker can impersonate a legal user to access the service. Nevertheless, previous authentication mechanisms seldom focus on how to prevent this threat but external attacks. Hence, we aim to design an authorization transfer protocol to eliminate this malicious threat, in which a smartphone has been used to help the access transfer. That is, a user can carry on network services without keying any secret information into the public computer once a switch from a laptop or mobile device to a public computer is needed. In particular, we have simulated the system to demonstrate the performance of the proposed mechanism. Moreover, the correctness of mutual authentication has been proved according to the AVISPA. The proposed method allows users to securely transfer their services to public access devices through their smartphones without disclosing their sensitive information.

Keywords: Authorization transfer, Confidentiality, Public access, Keylogger, Smartphone

1 Introduction

In recent decades, the provision of various online services through the Internet has led to an increasing expectation among users to access these services from anywhere. However, this convenience is accompanied by information security risks, as inadequate security measures can expose personal data such as account passwords, credit card numbers, and residential information to the Internet. Therefore, authentication mechanisms are essential for verifying the legitimacy of both service providers and users. Among them, the password authentication one [1-5] 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 can maintain the service access according to the comparison between the received authentication token and the one recorded in the database.

Nevertheless, researchers have pointed out that this simple mechanism might suffer from the stolen verifier attack and 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 [6-13]. Personal information and authentication tokens are kept in the card to prove the validity instead of the server database. Hence, the risk of stolen verifier attacks can be eliminated.

Subsequently, there have been 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 or impersonation attack. In [13], Song has proposed a secure password authentication mechanism which depends on the assumption that no one could dig information from the card, namely, the strong smart card. In addition, two-factor authentication schemes have been designed to enhance the entropy of verification token [14-18], in which the difficulty in compromising the system could be highly reinforced. Aside from a smart card, people often adopt the fingerprint as the other factor for securing personal information. The sampling process of fingerprint, however, is a tough problem in the implementation. It is due to the high sensitivity of cryptographic function. Thus, researchers start to apply a smartphone to be the second factor instead of the fingerprint. According to Groupe Speciale Mobile Association (GPMU) statistics in 2022, there are more than 4.3 billion smartphone subscribers around the world [19]. The adoption of smartphone does make sense while integrating an authentication mechanism. Specifically, 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.

With the explosive development of electronic com-

*Corresponding Author: Jung-San Lee; Email: leejs@fcu.edu.tw
DOI: https://doi.org/10.70003/160792642026012701005

merce and communication technologies, validity is not the only essential that these authentication mechanisms have to achieve. Many new challenges, including anonymity, un-traceability, efficiency, and resistance to new attacks, have come out in designing a novel authentication mechanism [12]. Nevertheless, common scenarios are often overlooked in the field of authentication mechanisms. Specifically, users frequently need to log into online systems using devices that are not their own, such as public computers or laptops belonging to others. Ensuring the protection of login information in these situations is particularly challenging. Public computers, available in places like campuses and libraries, pose significant security risks as attackers can install keylogging software to capture users' credentials [20]. Incidents at universities in the United States [28-29] have demonstrated the real dangers of this threat, with students installing keyloggers to steal staff credentials, alter grades, and access sensitive information, thereby compromising the rights and privacy of other students. Moreover, an attacker can scrap the memory to obtain the security information and launch an impersonation attack [21]. 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.

So far, most authentication mechanisms are designated to check the user validity and secure the transferring token. However, ignoring the real-world context of using public equipment has led to the occurrence of these aforementioned attacks. Thus, we aim to propose a new authority transferring mechanism, in which a user can use the mobile phone to switch a service to a temporary computer instead of entering any personal secret. Aside from providing secure authority transferring services, the contributions of this paper include:

1) Restriction: We design the service provider to be capable of controlling one-time access transfers, and it can prevent a malicious attacker from reusing the information to obtain the service.
2) Selectivity: We supply the strategy on the mechanism, and then the user can directly communicate with the temporary computer. There is no need for an additional device for connecting, such as a Wi-Fi access point, Bluetooth sensor, or SMS server.
3) Availability: We analyze the overhead of the computation, and implement the mechanism.
4) Security: We use the formal tool AVISPA [22], which has been used for validating robustness in numerous researches to prove the mutual authentication. Furthermore, we analyze various kinds of attacks, and present how the proposed mechanism can resist.

The rest of the paper is organized as follows. In section 2, we describe the environment of the system, and show the detail of the mechanism. Then, we analyze the achieving requirements, the performance with the simulation, and the security in section 3. Finally, we make conclusions and future work in section 4.

2 The Proposed Mechanism

In this section, we describe the proposed mechanism, including the environment setup, registration phase, and transference phase.

2.1 Environment Setup

In the proposed mechanism, the App Server (*AS*) is assumed to be the trusted third party, it can help a mobile user to transfer the service, and the user must have registered at *AS* before requesting. And *S* is a service provider, offering the service for registered users. Once a mobile user (*MU*) is located at a public space, he operates his mobile phone login to *AS* for requesting, and transfers the service from *S* to a temporary computer (*PC*). Below, we present transmission architecture, essential assumption with the mechanism, and the notations is defined in Table 1.

Table 1. Notations of new mechanism

Notation	Description
x	Master key of *AS*
ID_x	An identify of x
r_x	A random number is generated by x
N_x	A random nonce is generated by x
$h(\cdot)$	One-way hash function
$(M)_k$	XOR cipher algorithm (The key K and the message M are extend to same length, and the message is encrypted by exclusive-OR operation)
$[M]_k$	Symmetric-key algorithm (The message M is encrypted by secret the key K)
log	Log File (The message can request corresponding service from the service provider)

2.2 Transmission Architecture

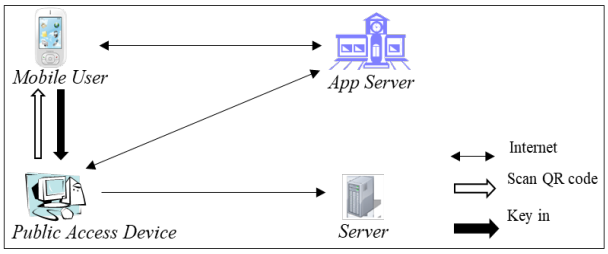


Figure 1. The architecture of transmission

Figure 1 describes the architecture of the proposed mechanism. A user *MU* interacts with *AS* via the Internet, and delivers the information to *PC* through the keyboard, and *PC* provides the information via Quick Response Code (QR code) for *MU*, the detail of the communication is expended in Section 2.4. Note that the communication between *MU* and *PC* is physical, the public space can lessen extra devices to connect with the user through QR code strategy, and Section 3.2 will describe computation overhead of QR code. Moreover, in our design, the user

communicates with *AS* directly, which could avoid a large amount of public devices indefinitely keeping the request.

2.3 Essential Assumption

In new mechanism, we assume a pre-share key key_{AS-S} which has been agreement on between *AS* and *S*, as Figure 2. And when transference phase, S can confirm the validity of delivering message via the key. Nevertheless, there is no directly communication between *AS* and *S* in our mechanism, it could reduce extra cost with *AS*.

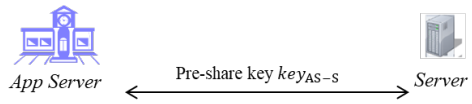


Figure 2. The flowchart of system setup

2.4 Mechanism Description

The proposed mechanism consists of two phases: the registration phase and the transference phase. *MU* must have registered at *AS* though the registration phase, and when the user is going to transfer the service, he shall operate the transference phase.

1) *Registration phase:* The flowchart of the registration phase is illustrated in Figure 3. This phase is invoked whenever the mobile user *MU* initially registers or re-registers to *AS*.

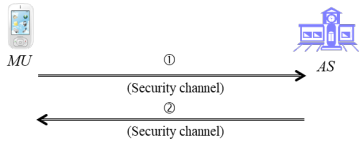


Figure 3. The flowchart of registration phase

Step 1:

MU sends his real identification ID_{MU} to *AS* via secure channel.

Step 2:

While *AS* receives registration request, it generates a random number r_{AS}, and computes $AID_{MU} = h(ID_{MU} || r_{AS})$. Next, it employs AID_{MU} and its master key x to calculate $key_{MU-AS} = h(AID_{MU} || x)$. Finally, *AS* delivers AID_{MU} and key_{MU-AS} to *MU* though secure channel.

2) *Transference phase:* After *MU* has registered at *AS*, he can request transferring the service to the public access device, and the flowchart of the transference phase is illustrated in Figure 4.

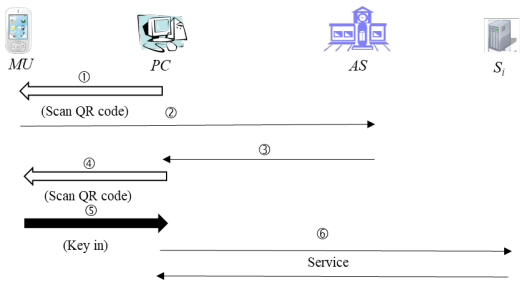


Figure 4. The flowchart of request service phase

Step 1:

PC provides its identification ID_{PC} via QR code.

Step 2:

Upon *MU* obtains ID_{PC}, he generates a random number r_{MU} and a random nonce N_{MU}. Next, *MU* employs N_{MU}, key_{MU-AS} to calculate the nonce key $h(key_{MU-AS} || N_{MU})$. And then *MU* uses the nonce key to encrypt ID_{PC}, ID_S, *log*, r_{MU}, $h(AID_{MU}||r_{MU}||ID_{PC}||ID_S||N_{MU})$.

Subsequently, *MU* sends $((ID_{PC}, ID_S)key_{MU-AS}, log, r_{MU}, h(AID_{MU}||r_{MU}||ID_{PC}||ID_S||N_{MU}))_{h(key_{MU-AS}||N_{MU})}$, AID_{MU}, N_{MU} to *AS*.

Step 3:

While *AS* receives the message from *MU*, it firstly confirms the validity of N_{MU}. Next, it computes the nonce key $h(h(AID_{MU}||x)||N_{MU})$ to decrypt receiving cipher-text. For checking the integrity of message, it calculates the hash value $h(AID_{MU}||r_{MU}||ID_{PC}||ID_S||N_{MU})$ to compare with the received one. If they are different, *AS* shall reject the request; otherwise, *AS* generates a random nonce N_{AS} and computes $h(AID_{MU}||r_{MU} + 1||N_{AS})$, After computing, it encodes the hash value and N_{AS} to QR code at selective strategy. The QR code forwards to *MU* via *PC*, and the user can confirm validity.

Subsequently, *AS* generates a random nonce N_1, and finds out the pre-shared key key_{AS-S}, which is correspond with ID_S. Then, it calculates $h(key_{AS-S} || N_1)$ to encrypt service information ID_{PC} and *log* based on the symmetric encryption function. For confidentiality of the message, *AS* computes an other key $h(ID_{MU}||r_{MU})$, and encrypts $[log, ID_{PC}]_{h(key_{AS-S}||N_1)}$, N_1.

Finally *AS* sends the message $[[log, ID_{PC}]_{h(key_{AS-S}||N_1)}, N_1]_{h(ID_{MU}||r_{MU})}$, $h(AID_{MU}||r_{MU} + 1||N_{AS})$, N_{AS} to *PC*.

Step 4:

Upon receiving the message from *AS*, *PC* exhibits the receiving QR code to *MU* and keeps the information $[[log, ID_{PC}]_{h(key_{AS-S}||N_1)}, N_1]_{h(ID_{PC}||r_{MU})}$ to wait for using.

Step 5:

After *MU* obtains the information $h(AID_{MU}||r_{MU} + 1||N_{AS})$, N_{AS} by scanning, he checks the validity of N_{AS}. And then, he calculates and compares $h(AID_{MU}||r_{MU} + 1||N_{AS})$ with the received one. If the hash value is valid, the user keys r_{MU} into *PC*; otherwise, the user need requests *AS* again.

Step 6:

Upon receiving the message from *MU*, *PC* computes $h(ID_{PC}||r_{MU})$ to decrypt keeping message $[[log, ID_{PC}]_{h(key_{AS-S}||N_1)}, N_1]_{h(ID_{PC}||r_{MU})}$. Then, it sends the decrypting message $[log, ID_{PC}]_{h(key_{AS-S}||N_1)}$, N_1 to *S*. While *S* obtains the information, it confirms the validity of N_1. If it is invalid, the procedure is terminated; otherwise, *S* computes $h(key_{AS-S} || N_1)$ to decrypt the message, and then it provides the corresponding service to *PC*. Simultaneously, the user locates with the public device, he can obtain the service on the device.

3 Analyses

Here, we discuss the requirements of the new mechanism, which are based on the symmetric encryption

function, one-way hash function, and XOR-cipher. The assumptions is are guaranteed as the follow.

(1) Symmetric encryption function $[\cdot]_K$

Given a plaintext P, it is easy to compute the cipher-text $C = [\cdot]_K$ with a symmetric key k. But it is computationally infeasible to gain P from C with lacking k [23].

(2) One-way hash function $h(\cdot)$

For a security one-way hash function $h(\cdot)$ and a message M, it is easy to compute the digest $y = h(M)$. Furthermore, $h(\cdot)$ can confirm the following properties [24]:

(i) Preimage resistance — Give a digest, it is computationally infeasible to find out the input (pre-image). In another word, it is difficult to find M such that $h(M) = y$ when y is known.

(ii) Second preimage resistance — It is computationally infeasible to find another input, which has the same digest; i.e. give a message M, it is hard to find another message $M'(\neq M)$ such that $h(M') = h(M)$.

(iii) Collision resistance — It is computationally infeasible to find two different inputs with the same digest; i.e. it is difficult to find M and M' such that $h(M') = h(M)$.

(3) XOR cipher $(\cdot)_k$

Given a plaintext P, it is easy to compute the cipher-text $C = M \oplus k$ with k. However, it is difficult to obtain P from C without k.

3.1 Restricted Access

In order to avoid repeatedly requesting the service from the attacker, the service provider need has authority to restrict with invalid access. In step 6 of the transference phase, S receives the request from PC, and confirms the validity of receiving nonce N_1. If it is valid, S employs its pre-share key key_{AS-S} and the nonce to calculate the key $h(key_{MU-AS}||N_{MU})$. Then, it decrypts the message to obtain the right access device, and sends the corresponding service to MU; otherwise, the provider will reject the access. Because there is no one can fake the encrypting message $[log, ID_{PC}]_{h(key_{AS-S}||N_1)}$ without the pre-share key key_{MU-AS}. Therefore, the service provider is able to control the access, even the attacker uses the key-logger program or scrapping the memory.

3.2 Selectivity

In our mechanism, we supply a selective strategy via QR code. PC can deliver information to the user via QR code in step 1 and 4 of the transference phase, and that overhead of the strategy is illustrated in Section 4. Following the strategy, the public space does not need to prepare the additional physical device, and the user directly communicates with public space. Furthermore, even if the attacker intercepts the information being sent, which includes only the identification and the digest, it is difficult to obtain any private information about the user due to the one-way hash function.

3.3 Privacy Protection

In our environment, it is high probability obtain the privacy information of the user (such as ID, password) form the temporary computers via key-logger program or scrapping memory, and the attacker cab impersonate a large user to access the server according to the information.

In the proposed mechanism, the user only keys the random number r_{MU} into the temporary computers. Even though a malicious attacker Eve obtained the message via the key-logger program, she cannot obtain any secret information about the user; On the other hand, storing information in the memory of the temporary computers, it is hard to obtain security information from the digest $h(ID_{MU}||r_{MU} + 1||N_{AS})$ under the assumption of one-way hash function. Hence, we can be sure the new mechanism can protect user privacy.

3.4 Anonymity

In the registration phase, the legal user offers his real identification ID_{MU} to AS via the secure channel. Then, the server generates a random number r_{AS} to calculate the anonymous identification $AID_{MU} = h(ID_{MU}||r_{AS})$. If a malicious attacker wants to get the real identification of the user, she must be failed. Because it is difficult to find the pre-image from the digest under the assumption of the one-way hash function. According to this design, the mechanism can avoid leaking the identity with the user.

3.5 Security

In the section, we show some common attacks, and analyze how the new mechanism can resist these attacks.

3.5.1 Replay Attack

Assume that a malicious attacker Eve intercepts delivering message from a legal user MU to AS and launches the replay attack. If she directly sends the intercepting message to AS, AS can easily perceive the attempt according to the freshness of the random nonce N_{MU}. Of course, she might replace N_{MU} with a valid nonce N_E for the attack. But even that, she cannot calculate the message digest $h(ID_{MU}||r_{MU}||ID_{PC}||ID_S||N_{MU})$ without r_{MU}, thus AS can compare the different from the digest. In addition, the digest and the random number r_{MU} are based on exclusive-OR encryption function, and encrypting key $h(key_{MU-AS}||N_{MU})$ is interacted with the nonce, it is different to get the pre-share key by the attacker [25].

On the one hand, Eve tires tries to send a forged digest to the user in order to obtain the main information r_{MU} in Step 4 of the transference phase. However, Eve cannot compute the valid message $h(ID_{MU}||r_{MU}||N_{AS})$. Hence, even though she intercepts the message $[log, ID_{PC}]_{h(key_{AS-S}||N_1)}$, N_1 to send to the service provider S, she still not obtains the service. Because the message is protected with the validity of random nonce N_1 and pre-share key key_{AS-S}. By the way, we offer the challenge-and-response to the digest, and MU can check whether r_{MU} is really received according to avalanche effect.

3.5.2 Server Spoofing Attack

If a malicious attacker Eve masquerade as AS to defraud MU, she must be failed. Because even though Eve can intercepts AID_{MU}, N_{MU}, she cab not get r_{MU} from the delivering message $(ID_{PC}, ID_S, log, r_{MU}, h(AID_{MU}||r_{MU}||ID_{PC}||ID_S||N_{MU}))_{h(key||N_{MU})}$ under the exclusive-OR encryption. Moreover, it is impassable to compute the key

$h(h(AID_{MU}||x)N_{MU})$ without the master key x of AS. As a result, Eve cannot generate a valid message $h(ID_{MU}||r_{MU}+1||N_{AS})$ to cheat MU under the assumption of one-way hash function. Thus, we ensure a malicious attacker is very difficult to impersonate AS, and the mechanism can resist the server spoofing attack.

3.5.3 Impersonation Attack

Providing that an attacker Eve can impersonate a large user request for the transferring service, it will infringe the rights of the user. However, even Eve intercepts the anonymous identity AID_{MU} form Step 1 of transference phase, she is unable to calculate $h(AID_{MU}||x||N_{MU})$ without the master key x under the one-way hash function. Hence, she cannot send a foraged message to cheat AS.

On the other hand, assume that Eve has registered at AS, she uses her session key key_{Eve-AS} and replaces with the identity AID_{MU} to generate a fake message ID_{PC}, ID_S, log', r_E, $h(AID_E||r_E||ID_{PC}||ID_S||N_E))_{h(key_{Eve-AS}||N_E)}$, AID_{MU} , N_E for impersonating. Even though the message digest is valid, AS will compare different after decrypting by the calculating key $h(h(AID_{MU}||x)||N_E$, and AS will reject the request. Hence, the new mechanism is able to withstand the impersonation attack.

3.5.4 Off-line Guessing Attack

If an attacker Eve aims to obtain r_{MU}, she may intercepts delivering digest $h(AID_{MU}||r_{MU}+1||N_{AS})$. But it need to spend lots of time for guessing the random number r_{MU} based on the one-way hash function. Maybe Eve obtains r_{MU} after a long time, and she decrypts the keeping message to send $[log, ID_{PC}]_{h(key_{AS-S}||N_1)}$, N_1 to S. But according to the validity of N_1, S will reject the request from Eve. Even though Eve can generate a valid nonce N_E and replace N_1. It is hard to compute $h(key_{AS-S}||N_E)$ without key_{AS-S} under the assumption of one-way hash function. Thus, the attacker cannot launch the off-line guess attack.

3.5.5 Stolen Verifier Attack

In reason year, some papers use the password table to support the verification, but that would launch the stolen verifier attack. In our mechanism, rather than AS recodes information with the user, it employs its master key x to support verification. Thus, the mechanism can reduce the overhead of the database, and simultaneously resist the stolen verifier attack.

3.5.6 Key-log Attack

Assume that a malicious attacker Eve had installed the key-logger program into the public access device, and intercepts the keying information r_{MU} after the communication. However, while she employs r_{MU} to decrypt the cipher-text $[log, ID_{PC}]_{h(key_{AS-S}||N_1)}$, $N_1]_{h(ID_{PC}||r_{MU})}$, and send to S. The provider would confirm the nonce N_1 is used. Even though Eve replaces with the valid nonce, S still perceive the attempt according to validity of key_{AS-S}. Hence, we can conclude the proposed mechanism is capable of preventing the key-log attack.

3.5.7 Ram-scraping Attack

In 2009, Halderman et al. proposed an attacker can scrap the memory to obtain the security information [21]. This attacker can impersonate a legal user to access the service, or use the information to cheat the other service providers into getting different services. However, even though Eve obtains the information $[log, ID_{PC}]_{h(key_{AS-S}||N_1)}$, N_1 from scraping the memory of the temporary computer, she still obtain random number, hash value, and encrypting message, there is no any secret information with the user. Even Eve sends the request message to the service provider, S can confirm the validity of N_1, and key_{AS-S}, and successfully resist the wrongful request. Thus, the mechanism can withstand ram-scraping attacks.

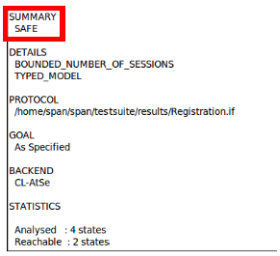
3.5.8 Mutual Authentication

In this section, we utilize the popular formal proof validation tool Automatic Validation of Internet Security Protocols and Applications (AVISPA) [22] to ensure protocol security. AVISPA is an automated, push-button tool for the formal validation of internet security protocols. It covers all security protocols in the first five layers of the OSI model. Furthermore, AVISPA encompasses IETF security specifications, making it a widely utilized tool in numerous research studies. The version of AVISPA is modeled by Security Protocol Animator version 1.6 (SPAN 1.6) on Ubuntu 10.10-light.

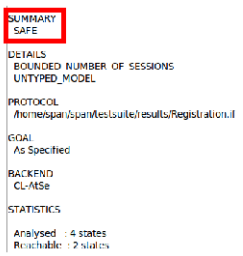
AVISPA [22] utilizes High-Level Protocol Specification Language (HLPSL) to analyze protocol security. The simulation environment categorizes protocols into roles, environments, goals, and sessions. This environment replicates the transport environment detailed in subsections 2.1 and 2.2. The role corresponds to the MU, PC, $App\ Server$, and S_i.

In the security analysis, AVISPA [22] uses different security modules to simulate various attacks, such as replay attacks, user simulation and server spoofing, to assess whether the protocol meets the validation criteria. AVISPA simulates message transmission in protocols, to ensure inclusiveness, two modules, the Constraint-Logic-based Attacker Searcher (CL-AtSe) and the On-the-Fly-Model-Checker (OFMC), are used for validation. CL-AtSe checks for legitimacy and validation in a limited number of sessions [26]. Its modular design facilitates the integration of operator attributes such as exponentiation and exclusive-or [27]. Specifically, CL-AtSe consists of Typed models using all parameters, Untyped models with generic parameters, and Verbose models detailing potential attacks.

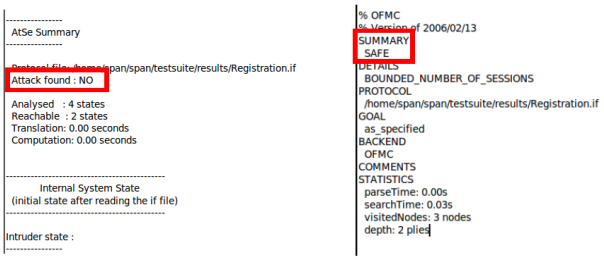
At the same time, OFMC is used to analyze security protocols through lazy, demand-driven search. The lazy intruder technique ensures the identification of all possible attacks. In addition, the constraint differentiation technique is a holistic search approach that reduces the analysis search time [27].



(a) Typed model

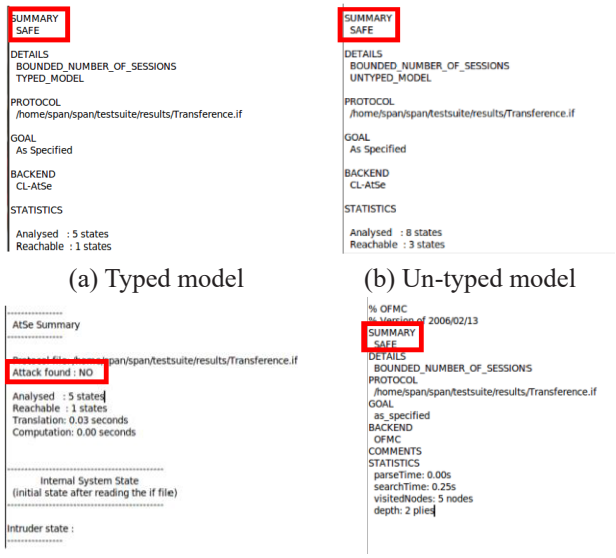


(b) Un-typed model



(c) Verbose mode (d) OFMC

Figure 5. The result of security analysis in registration phase



(a) Typed model (b) Un-typed model

(c) Verbose mode (d) OFMC

Figure 6. The result of security analysis in transference phase

Each stage of protocol security is verified independently, as shown in Figure 5 (a-d) to Figure 6 (a-d). The red-boxed part of the figure indicates whether the test result confirms security or not. Obviously, each stage meets the security criteria.

4 Performance Discussion

In this section, we analyze the performance of our mechanism. The cost of the computation is shown in Table 2.

Table 2. The computational overhead of the mechanism

	MU	*PC*	*AS*	*S*
Register phase			$2T_h$	
Request service phase	$2T_h+$ $1T_{xor-c}$	$1T_{sys}+$ $1T_h$	$1T_{sys}+$ $6T_h+$ $1T_{sys}$	$1T_{sys}+$ $1T_h$

T_h: Hash; T_{xor-c}: XOR-cipher; T_{sys}: Symmetric

To improve the availability, we conduct experiments to simulate the mechanism. In the experiment, we shall not simulate with the service provider, and the computational overhead of which can refer *PC*. The environment of the experiment is illustrated in Table 3.

Table 3. The experimental situation of the new mechanism

Mobile user (Sony ion L28i)	CPU	Qualcomm S3 MSM8260 1.5 GHZ
	RAM	1 GB
	OS	Android 4.1.3
Public access device	CPU	Inter(R) Pentium E2205 2.49 GHz
	RAM	4 GB
	OS	Window 8.3
App server	CPU	Inter(R) Core(TM) i7 930 2.80 GHz
	RAM	8 GB
	OS	Windows 7

As below, we perform each simulation 100,000 rounds. In the experiment, we set the key size of RSA is 128 bits, the plain-text is no longer than 128 bits, the one-way hash function use SHA-1, The performance is shown in Table 4. Furthermore, for demonstrating the new mechanism can be applied to mobile phone, we modify the frequency of CPU from 0.384 to 1.512 in order to, and the result is illustrated in Figure 7.

Table 4. The computational overhead of the mechanism based on experimentation

(ms)		Average	(min , max)	σ
MU	*Enc_req*	0.832	(0.549 , 41.840)	2.066
	Ver_hash	0.131	(0.061 , 37.659)	0.825
	subtotal	0.963	(0.610 , 79.498)	2.891
PC		0.046	(0.043 , 0.720)	0.024
AS		0.071	(0.067 , 20.336)	0.073
Total		1.080	(0.720 , 100.554)	2.987

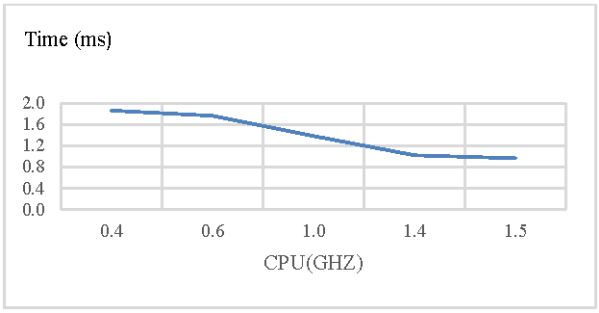


Figure 7. The performance of mobile user based on different CPU frequencies

According to the result, we know the proposed mechanism can fluently operate on devices with lower computational capabilities, let alone on modern smartphones, where it can complete verification more efficiently. Next, we provide the computational cost of *MU* and *AS* in QR code strategy in Table 5. Note that the mobile user needs to decode two times in the mechanism. Moreover, we show

the overhead of the computation based on different CPU frequencies in Figure 8.

Table 5. The computational overhead of the QR code strategy based on experimentation

(ms)	Average	(min , max)	σ
Encoding (*MU*)	4.271	(3.967 , 33.661)	1.521
Decoding (*AS*)	0.715	(0.566 , 17.155)	0.104
Total	9.257	(8.500 , 84,477)	3.145

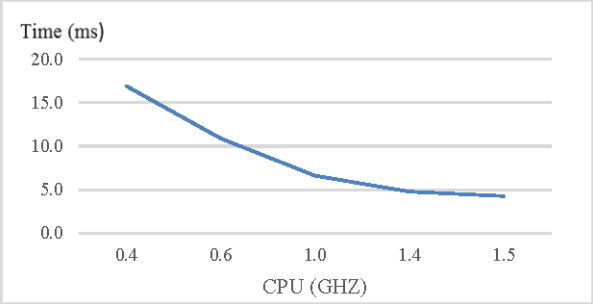


Figure 8. The performance of QR code strategy based on different CPU frequencies of mobile user

From the result, it can complete the transference through the intelligent mobile phone. Although the intelligent mobile phone shall spend more time, the system does not need an additional device environment.

Finally, we show the computational cost of the mechanism with the QR code strategy, which is based on different CPU frequencies as Figure 9. And the system can select whether to use according to the power of the mobile phone.

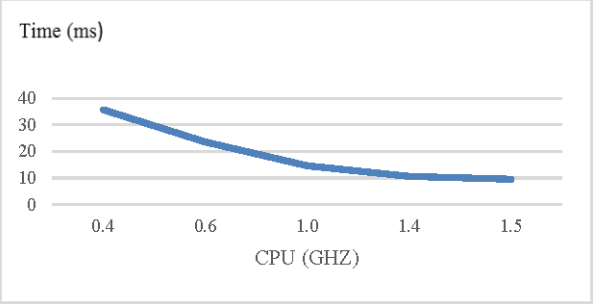


Figure 9. The performance of the mechanism with QR code strategy based on different CPU frequencies of mobile user

5 Conclusions

In this paper, we have presented the weakness of the traditional authentication mechanisms, the attacker can easily get the user privacy from temporary computers with the key-logger and ram-scraping. For this motivation, we propose a new mechanism that can solve the problem. The user can securely transfer their own service to the public access device once via a smart phone, and there is no leakage with the security information of the user. Moreover, we supply a strategy to provide the selectivity with the user. Finally, we implement the mechanism to prove the performance, and analyze the security to show that can prevent the malicious attacks.

References

[1] A. Evans, Jr., W. Kantrowitz, E. Weiss, A user authentication scheme not requiring secrecy in the computer, *Communications of the ACM*, Vol. 17, No. 8, pp. 437-442, August, 1974.
https://doi.org/10.1145/361082.361087

[2] H. Feistel, W. A. Notz, J. L. Smith, Some cryptographic techniques for machine-to-machine data communications, *Proceedings of the IEEE*, Vol. 63, No. 11, pp. 1545-1554, November, 1975.
https://doi.org/10.1109/PROC.1975.10005

[3] R. Lennon, S. Matyas, C. Meyer, Cryptographic authentication of time-invariant quantities, *IEEE Transactions on Communications*, Vol. 29, No. 6, pp. 773-777, June, 1981.
https://doi.org/10.1109/TCOM.1981.1095067

[4] C.C. Chang, S. J. Hwang, Cryptographic authentication of passwords, *IEEE 25th Annual 1991 International Carnahan Conference on Security Technology*, Taipei, Taiwan, 1991, pp. 126-130.
https://doi.org/10.1109/CCST.1991.202203

[5] L. Harn, D. Huang, C. S. Laih, Password authentication using public-key cryptography, *Computers & Mathematics with Applications*, Vol. 18, No. 12, pp. 1001-1017, 1989.
https://doi.org/10.1016/0898-1221(89)90028-X

[6] T. C. Wu, Remote login authentication scheme based on a geometric approach, *Computer Communications*, Vol. 18, No. 12, pp. 959-963, December, 1995.
https://doi.org/10.1016/0140-3664(96)81595-7

[7] W. Yang, S. Shieh, Password authentication schemes with smart cards, *Computers & Security*, Vol. 18, No. 8, pp. 727-733, 1999.
https://doi.org/10.1016/S0167-4048(99)80136-9

[8] M. L. Das, A. Saxena, V. P. Gulati, A dynamic ID-based remote user authentication scheme, *IEEE Transactions on Consumer Electronics*, Vol. 50, No. 2, pp. 629-631, May, 2004.
https://doi.org/10.1109/TCE.2004.1309441

[9] Y. Y. Wang, J. Y. Liu, F. X. Xiao, J. Dan, A more efficient and secure dynamic id-based remote user authentication scheme, *Computer Communications*, Vol. 32, No. 4, pp. 583-585, March, 2009.
https://doi.org/10.1016/j.comcom.2008.11.008

[10] K. H. Yeh, C. Su, N. W. Lo, Y. Li, Y. X. Hung, Two robust remote user authentication protocols using smart cards, *Journal of Systems and Software*, Vol. 83, No. 12, pp. 2656-2565, December, 2010.
https://doi.org/10.1016/j.jss.2010.07.062

[11] S. Q. Cao, W. R. Liu, L. L. Cao, X. He, Z. Y. Ji, An Improved Authentication Protocol Using Smart Cards for the Internet of Things, *IEEE Access*, Vol. 7, pp. 157284-157292, October, 2019.
https://doi.org/10.1109/ACCESS.2019.2949649

[12] Y. Chen, W. Kong, X. Jiang, Anti-Synchronization and Robust Authentication for Noisy PUF-Based Smart Card, *IEEE Access*, Vol. 7, pp. 142214-142223, September, 2019.
https://doi.org/10.1109/ACCESS.2019.2944515

[13] R. Song, Advanced smart card based password authentication protocol, *Computer Standards & Interfaces*, Vol. 32, No. 5-6, pp. 321-325, October, 2010. https://doi.org/10.1016/j.csi.2010.03.008

[14] D. S. Wang, J. P. Li, A new fingerprint-based remote user authentication scheme using mobile devices, *International Conference on Apperceiving Computing and Intelligence Analysis*, Chengdu, China, 2009, pp. 65-68.

[15] B. T. Hsieh, H. T. Yeh, H. M. Sun, Cryptanalysis of a fingerprint-based remote user authentication scheme using smart cards, *IEEE 37th Annual 2003 International Carnahan Conference on Security Technology*, Taipei, Taiwan, 2003, pp. 349-350. https://doi.org/10.1109/CCST.2003.1297584

[16] J. Xu, W. T. Zhu, D. G. Feng, Improvement of a fingerprint-based remote user authentication scheme, *International Conference on Information Security and Assurance*, Busan, Korea (South), 2008, pp. 87-92. https://doi.org/10.1109/ISA.2008.62

[17] D. Wang, P. Wang, Two Birds with One Stone: Two-Factor Authentication with Security Beyond Conventional Bound, *IEEE Transactions on Dependable and Secure Computing*, Vol. 15, No. 4, pp. 708-722, July-August, 2018. https://doi.org/10.1109/TDSC.2016.2605087

[18] A. Derhab, M. Belaoued, M. Guerroumi, F. A. Khan, Two-Factor Mutual Authentication Offloading for Mobile Cloud Computing, *IEEE Access*, Vol. 8, pp. 28956-28969, February, 2020. https://doi.org/10.1109/ACCESS.2020.2971024

[19] M. Shanahan, K. Bahia, The State of Mobile Internet Connectivity 2023, *Groupe Speciale Mobile Association*, October, 2023.

[20] S. Sagiroglu, G. Canbek, Keyloggers, *IEEE Technology and Society Magazine*, Vol. 28, No. 3, pp. 10-17, Fall, 2009. https://doi.org/10.1109/MTS.2009.934159

[21] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, E. W. Felten, Lest we remember: cold boot attacks on encryption keys, *Communications of the ACM*, Vol. 52, No. 5, pp. 91-98, May, 2009. https://doi.org/10.1145/1506409.1506429

[22] European Community under the Information Society Technologies Proframme, Deliverable D2.1: The High Level Protocol Specification Language, *Automated Validation of Internet Security Protocols and Applications*, IST-2001-39252, August, 2003.

[23] H. Delfs, H. Knebl, Symmetric-key encryption, *Introduction to Cryptography*, Springer, 2002, pp. 11-12.

[24] A. J. Menezes, P. C. Van Oorschot, S. A. Vanstone, Handbook of applied cryptography, CRC Press, 1997.

[25] C. E. Shannon, Communication theory of secrecy systems, *The Bell System Technical Journal*, Vol. 28, No. 4, pp. 656-715, October, 1949. https://doi.org/10.1002/j.1538-7305.1949.tb00928.x

[26] L. Viganò, Automated Security Protocol Analysis with the AVISPA Tool, *Electronic Notes in Theoretical Computer Science*, Vol. 155, pp. 61-86, May 2006. https://doi.org/10.1016/j.entcs.2005.11.052

[27] M. Turuani, The CL-Atse Protocol Analyser, in: F. Pfenning (Eds.), *Term Rewriting and Applications RTA 2006*, Vol. 4098, Springer, 2006, pp. 277-286. https://doi.org/10.1007/11805618_21

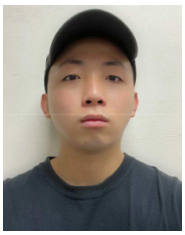
[28] U. Amir, Student Arrested for Using Keylogger and Changing Grades 90 Times, *Hackread*, November, 2017.

[29] L. Vaas, Student Jailed for Using Keylogger to Up His Exam Marks, *Sophos News*, April, 2015.

Biographies

Yun-Yi Fan is pursuing her PhD degree in information engineering and computer science in Feng Chia University, Taichung, Taiwan. Her current research interests include information security and blockchain applications.

Chung-Wei Kuo received his Ph.D. degree in Ph.D. program of Electrical and Communications Engineering in Feng Chia University, Taichung, Taiwan in 2016. His current research interests include information security and wireless communications.



Tzu-Hao Chen is currently pursuing his PhD degree in information engineering and computer science in Feng Chia University, Taichung, Taiwan. His current research interests include network security and blockchain applications.

Chia-Hung Chang received his MS degree in information engineering and computer science in Feng Chia University, Taichung, Taiwan in 2014. His current research interests include wireless communications and network security

Jung-San Lee (Senior Member, IEEE) received his Ph.D. degree in computer science and information engineering in 2008 from National Chung Cheng University, Chiayi, Taiwan. Since 2023, he has worked as a distinguished professor in the Department of Information Engineering and Computer Science at Feng Chia University. He is also the vice president and director general of Cybersecurity Technology Institute, Institute for Information Industry. His current research interests include cybersecurity, electronic commerce, and blockchain.