Journal of Basrah Researches ((Sciences)) Vol. (47). No. 1 (2021)

Available online at: https://www.iasj.net/iasj/journal/260/issues

ISSN -1817-2695

Received date: 6-5-2021 Accepted date: 26-7-2021 Available online date: 31-12-2021

Design a Strong Scheme to Resist Phishing Attack

Mustafa H. Hashim[1], Ali A. Yassin[2]

[1,2]Basrah University, College of Education for Pure Science, Computer Science Dept.

E-mail : pgs2189@uobasrah.edu.iq, ali.yassin@uobasrah.edu.iq

Abstract

Currently, the number of URLs that lead to increase risks for defrauding authorized users and then the attackers try to gain illegal access to user's accounts over a malicious attack is increasing exponentially. The most dangerous attack that deals with the URL is phishing. This attack tries to access authorized accounts unlawfully way. The finances services, commercial, e-banking are followed by this type of attack because of the sensitive information they owned. In this paper, we present a secure verification scheme that has the facility to prevent the above malicious issues. Moreover, the proposed scheme will withstand well-known cyberattacks such as impersonation and MITM attacks. The proposed scheme also includes security features such as strict authentication, forward secrecy, and user identity anomaly. The security analysis and the experimental results showed the strongest of the proposed scheme compared with other related works based on the Scyther tool. Finally, our proposed scheme needs 1152 communication bits in the verification phase.

Keywords: phishing attack, Scyther, Levenshtein distance, Cyber-attacks, formal security analysis.

I. Introduction

Modern information technologies have recently received a lot of attention, but they are vulnerable to threats such as protection, malicious attacks, human error, and spam. [1]. The security challenges are considered one of the most serious threats to information technology and its applications The malicious attack are recommended to achieve a targeted investigation into computer systems and technology-dependent businesses. [2]. The primary objectives of these attacks are to intercept machine code, modify or erase data servers, and gain unauthorized access. There are several common malicious attacks such as Phishing attacks, Man-in-the-Middle (MITM) attacks, social engineering attacks, replay attacks, and denial-of-service attacks are the most common types of attacks. The phishing attack is a type of malicious attack that aims to steal information from users, such as passwords, social security numbers, credit card numbers, and login credentials through a fake URL[3, 4]. The fake URL can cause the detecting sensitive data of the victim, installation of malware, financial loss for victims, and put the organization's data at risk[5]. It is possible of entering a forged HTTP/HTTPS web server into the global domain name system (DNS) for sending messages to the legal user. Figure 1 explains the year-by-year phishing attacks from 2013 to 2019. So, when a phishing user is connected to the Internet, the forged HTTP/HTTPS seems to be a legitimate web server. Figure 2 shows the phishing attack scenario.

In this paper, we present a secure verification scheme to prevent phishing attacks and other cyber-attacks such as MITM, Replay, DoS/DDoS, Insider attacks. Furthermore, the proposed scheme proves formally using the Scyther tool that gives us a guarantee that the safety message exchanging mechanism between the legitimate parties. Lastly, we practically implement our work using the programming languages such as PHP, MySQL, JavaScript.

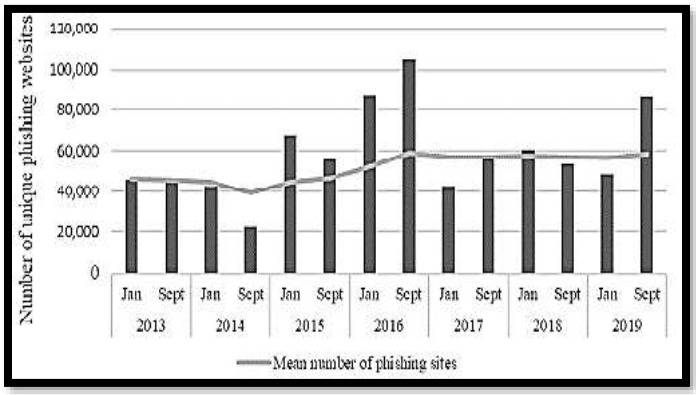


Figure 1. The phishing attacks on the websites between 2013 - 2019

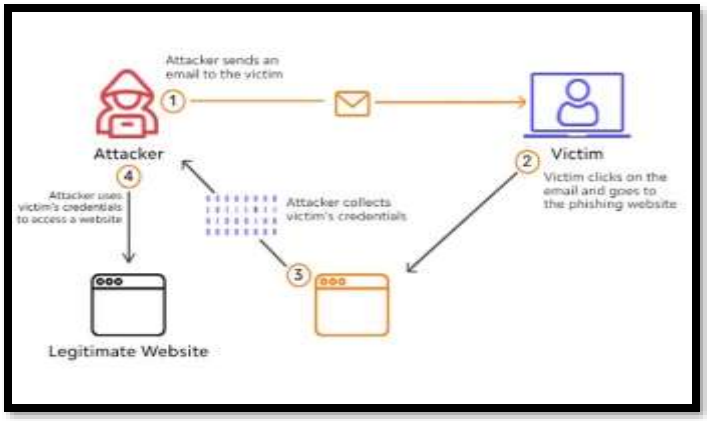


Figure 2. The phishing attacks scenario

The remaining sections of this paper are Section II: shows the related works. Section III: explains the proposed work. Section IV: shows the security analysis and the Implementation

II. Related work

***Huang et al.*[6]** established a method for avoiding online phishing attacks by redirecting visitors to legitimate websites using a site signature based on a combination of image and text-based attributes to retrieve the true URL. Their strategy has been hampered by the fact that signature construction allows phishers to use their attacks if the web page is missed.

***Bojjagani et al.* [7]** proposed a verification protocol to avoid phishing attacks in the mobile payment system using (ECDSA). Their protocol relies on an authentication server to give the user a nonce message and verify the validity of the user's signed data. To stop phishing attacks, an authentication server sends signed context information to the legitimate bank. [8]. The attacker imitates the protocol's actions by sending a bogus connection to the user and then sending a nonce to fool the user into thinking it's coming from the authentication server. Moreover, this work suffers from resisting impersonate attacks and needs strong verification. and also needs an anomaly of the user's identity property.

***Roy et al.* [9]** proposed a structure to use mobile-based authentication in a cloud computing. In this structure, they proposed a universal subscriber identity module (USIM) based on the identity verification technique. The USIM used as a main individuality to begin the authentication evolution. But in the event of the user's device become stolen, authentication will get unusably, and the whole process will get canceled[10].

***Lin et al* [11]** presented a protected scheme in the smart knowledge application in the cloud environment. The user is registered to the authentication server based on his ID. In their scheme, the user sends the hash of the password to the authentication server in the symmetrically encrypted form. The authentication server decrypts the password and obtains its hash value. Their scheme was secure against MITM attacks. However, since the users share the passwords, they are vulnerable to phishing attacks.

***Rose et al* [12]** proposed a strategy known as password hash ($PwdHash$) that can makes a different passwords for every website when the user uses the identical password for all these web-sites. If an attacker phishes a user's password, he will use it to log into any of the user's accounts. Their scheme aims to use $PwdHash()$ for making the current password is probabilistic. The mechanism of works realizes by applying $pwd' = pwdHash(pwd, seed)$ and sends the pwd' instead of pwd to the current website. The value of $seed$ is a unique parameter related to the website's domain name. Therefore, the attacker cannot phish the user's password to make illegal access to the user's websites.

Although the proposed scheme works perfectly, the attacker's goal is not to steal the user's password, he aims to steal the user's confidential data when he visits the attacker's website instead of a real website.

***Munivel et al.* [13]** proposed an authentication scheme to provide security in the mobile cloud environment, this scheme contains three parties (cloud user (*U*), cloud service provider (*CSP*), and trusted third party (*TTP*)). Their scheme has three phases. The first one creates a group called *G* and its members. The TTP shares the elements of the group with the communication entities. The second one handles the registration of *U* and *CSP* with the *TTP*. The last one checks the authority of *U* and *CSP* to achieve mutual authentication. Their scheme needs a strong verification because the *CSP* sends a nonce in a plain form that leads to a phishing attack by simulating the *CSP* behaviors. The attacker can impersonate the *CSP* and sends a nonce to *U* to redirect him to his malicious server.

***Alzuwaini et al.*[14]** proposed a verification scheme to avoid phishing attacks and other malicious attacks. In their scheme, they use an authentication server to verify the signed messages to determine the authority of the user. Additionally, they

employed a Secure index file to detect the phishers. Their scheme was analyzed in both formal and informal security analysis based on the scyther tool.

***Okunoe et al.*[15]** developed an anti-phishing approach using an advanced heuristic technique. In this approach, when a fishy website was discovered, the blacklist was immediately updated by inserting the website's domain name into this list. If a valid website is identified, the white list will update in the same manner. Additionally, when the user visits a website, this approach was first checked if the website was a phishing website or not and given permission to access the same website accordingly. This approach suffers from user privacy and needs security analysis against well-known cyberattacks such as MITM, reply, impersonate, insider. By the way, this approach requires more time in the computation/communication cost because it is needing access to the database for each login phase.

In this paper, we propose a secure verification scheme based on a symmetric encryption mechanism. The proposed scheme has many security merits like user's identity anomaly, key administration and resists the well-known attacks such as impersonate, phishing, replay, DoS, and device stolen attacks. Finally, our work is verified formally based on the scyther tool.

Table 1. The notation used in the proposed scheme

Character	*Description*
VI	User information that's needs to access the community site
U_i	The user
ID_i	User's unique Identity
SP_i	Service Provider Server
$SP_{i_{id}}$	Service Provider identity
$SP_{i_{URL}}$	Service Provider's domain name
TTP	Trusted Third Party
$S_{k_{U_i}}$	User's shared key
SK_{C_i}	Service Provider's shared key
LD	*Levenshtien Distance*
WL	White List
T_i	Time Stamp
DNS	Domain name system
M_i	Communication Message
f_i	Authentication flag
$\mathbb{Z}_q^*$	Group of q order and the elements are relatively prime with the order of a group
$\|\|$	Concatenation function
OTP	One Time Password

III. The Proposed Scheme

Our proposed scheme presents a safe verification mechanism to eliminate the Man-In-The-Middle (MITM), Social engineering, and Phishing attacks. Our work contains three legitimate parties: The User (U_i), The service provider Server (SP_i), The Trusted Third-Party Server (TTP). Moreover, the proposed scheme consists of three main phases: Initializing, User/Service Provider Registration, and Verification Phases. Figure.3 explains the structure of our work. The user interacts with the authorized Service Provider server such as University, Bank, Hospital based on his device included personal information such as Master card number, password. The service provider's server deals with real users (U_i) via legitimate web pages. After that, the service provider redirects the user to the *TTP* to confirming his entreaty and the service provider's URL. Furthermore, the *TTP* generates and sends an encrypted One Time Password (OTP) to the user (U_i). Lastly, the user checks the legitimacy of a *TTP* server by decrypting the VC. Additionally, our proposed scheme creates $White\ List(WL)$ contained the valid URLs of approved service providers to prevent phishing and malicious attacks.

Table 1 shows the main abbreviations used in our proposed scheme. Figure 3 explains our proposed scheme.

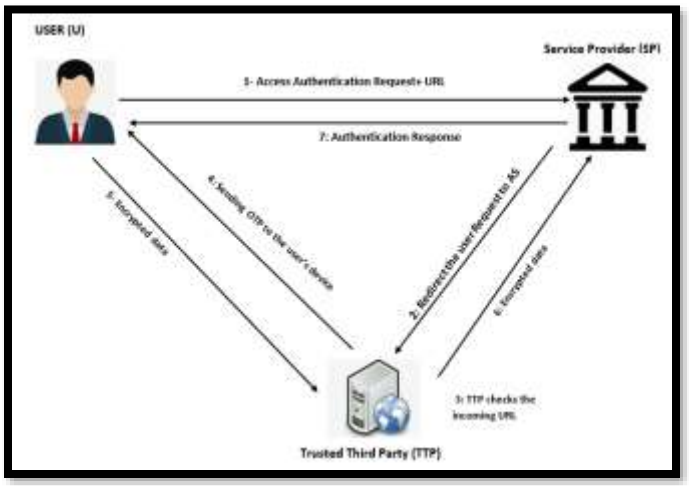


Figure 3. The Structure of Our Proposed Scheme

The main phases are:

1- Setup Phase:

In this phase, TTP generates a $White\ List$ (WL) contains the legal Domain Names of an authorized Service Providers.

2- Registration Phase

2.1 Community Registration Phase

1. In this phase, the SP_i enroll his data to TTP such as identity ($SP_{i_{id}}$), URL $\left(SP_{i_{URL}}\right)$.
2. Depended on receiving the information of SP_i, TTP compares $SP_{i_{URL}}$ with the WL based

on $Levenshtein\ distance(LD)$ if $LD \notin [1,2]$, TTP rejects this Service provider; Otherwise, he inserts the $SP_{i_{URL}}$ to the WL and completes this phase.

3. The TTP generates and sends the Service Provider's shared key SK_{SP_i} to the SP_i.

The arrangement of the Service provider's information in the TTP is explained in Table 2.

Table 2. Community information in TTP

#	$C_{i_{id}}$	$C_{i_{URL}}$	SK_{C_i}
1	2116	www.Amazon.com	43
2	3324	www.Al-Taif.com	25
3	6568	www.Master-card.com	76
4	7568	www.FBI.com	90

2.2 User Registration Phase

1. In this phase, the U_i enrolls his information and gives (Identity (ID_i), Personal identifying Number (PIN_i) to the SP_i. After that, SP_i checks the ID_i in his records, if not occur go to step 2; Otherwise, terminate this phase.
2. The SP_i forwards the ID_i to TTP for obtaining the shared key $(S_{k_i} \in \mathbb{Z})$ and then computes anomaly of user's identity $ID'_i = H(ID_i||C_{i_{id}})$.
3. TTP sends (S_{k_i}) to the U_i and sends ID'_i to SP_i.

Figure 4 shows the steps of the user registration phase.

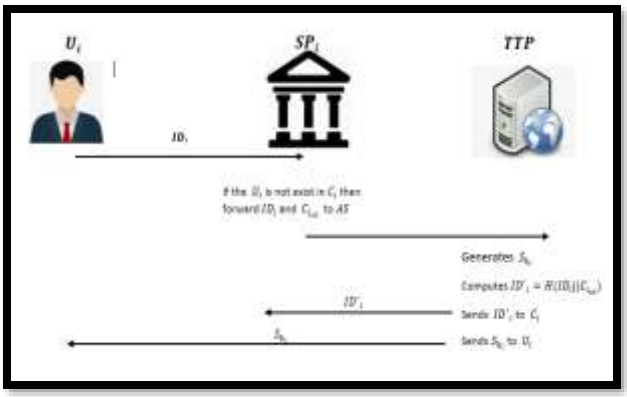


Figure 4. User Registration Phase

3. Verification Phase

In this phase, U_i follows the bellows steps when he receives the URL via his e-mail and then tabs on the received URL of the community for using the services of the URL. However, U_i sends $M_1 =< ID_i, URL >$ to SP_i.

$$SP_i \leftarrow U_i : M_1$$

1- Upon receiving the user's information M_1, SP_i checks the identity of U_i, if holds; SP_i sends $M_2 =< M_1, SP_{i_{id}}, ID'_i >$ to TTP over a secure connection channel. $TTP \leftarrow SP_i : M_2$.

2- Upon receiving M_2, TTP sets $f_i = 0$ and then restore the URL from M_2 to compare with the SIF depending on $Levenshtein\ distance(LD) = \forall j\ LD(URL, WL[j])$, If $LD \notin [1,2]$, AS sets $f_i = 1$. After that, he checks the authority of U_i by comparing ID'_i, if the result doesn't match then TTP terminates the current phase; Otherwise, TTP generates vOTP which consists of four digits. Additionally, TTP encrypts OTP using $E = Enc_{S_{k_i}}(OTP)$ then sends $M_3 =< E, TTP_{id}, SP_{id} >$ to the U_i.

$$U_i \leftarrow TTP : M_3.$$

3- Upon receiving M_3, the U_i perform the following steps:

- Set $VI =< URL,\ IP,\ SP_{id}, \dots etc >$.
- Compute $OTP' = Dec_{S_{k_i}}(E)$ and set $M_4 = (UI\ ||OTP')$.
- Encrypt $M_5 = Enc_{S_{k_i}}(M_4)$.
- Send M_5 to the TTP

4- TTP decrypts the received M_5using the shared key of the U_i using $M_5 = Dec_{S_{k_i}}(M_5)$

5- Compare $OTP \stackrel{?}{=} OTP'$, if they are a match, TTP informs SP_i via an encrypted message contained $M_6 = (ID_i||SP_{i_{id}}||T_1||f_i)$ where T_1 is a time of the verification process as follows:

$$M_6 = Enc_{SK_{SP_i}}(M_6)$$

And sends M_6 to the SP_i.

6- In the time T_2, SP_i receives M_6 then decrypts it using $M_6 = Dec_{SK_{C_i}}(M_6)$ and then Restores f_i from M_5 and compare $f_i \stackrel{?}{=} 1$, if they are equal then SP_i extract T_1 to compare with T_2 to determine the validity of login duration time, if $(T_1 - T_2 \leq \Delta T)$, SP_i response to the U_i. Otherwise, SP_i terminates the current session.

IV. Security Analysis and the Practical Implementation

1) Formal Security Analysis with the Scyther Tool:

Scyther is an important tool using for formal analysis. It works under the assumption that: an adversary must know the decryption key to obtaining the original message of the encrypted message. Scyther tool has many merits: 1) It uses an unbounded model to check a variety of security approaches (like authentication, verification, access control), 2) It grants permission to all possible behaviors, such as malware activity, to test the validity of a proposed scheme. Any proposed

scheme should be written in the security protocol description language (SPDL), which defines protocols and schemes, as well as support expressions for encryption, decryption, and signing, and also sending and receiving events.[16].

We write our proposed approach in SPDL and show the results in the cases of (Automatic Claim) and (Verification Claim). Figures 5(a, b), while figure 6 views the SPDL code in the scyther tool.

a. The verification Claim

b. The verification auto Claim

Figure 5. Verification results in scyther tool

/*Preventing Phishing attack*/

/*SPDL Analyser*/

```
usertype TTP,URL,Link,U,Uid,SP,SPid,U,TTPid,CI,VI,Dessision;
secret prk:Function;  // semetric key of U
secret prk2:Function;  // semetric key of SP
usertype SuccAuth;  //login completion
usertype T1;  //duration time
protocol DetectPhish(SP,TTP,U)
role U
const VCu,VCttp;
send_1(U,SP,(Uid,URL));
read_3(TTP,U,(TTPid,VCttp));
send_4(U,TTP,{{UI,VCu}prk(U)});
read_6(SP,U,(SuccAuth));
claim_U2(U,Secret,VCu);
claim_U3(U,Secret,VCttp);
claim_U4(U,Secret,Uid);
claim_U4(U,Secret,SuccAuth);
role SP
const VC,VCttp;
read_1(U,SP,(Uid,URL));
send_2(SP,TTP,(Uid,SPid,URL));
read_5(TTP,SP,{{UI,SPid,T1,Dessision}prk2(SP)});
send_6(SP,U,(SuccAuth));
claim_SP1(SP,Secret,UI);
claim_SP2(SP,Secret,T1);
}
role TTP
{
const VCu,VCttp;
send_3(TTP,U,(TTPid,VCttp));
read_2(SP,TTP,(Uid,SPid,URL));
read_4(U,TTP,{{VI,VCu}prk(U)});
send_5(TTP,SP,{{UI,SPid,T1,Dessision}prk2(SP)});
claim_TTP1(TTP,Secret,VCu);
```

claim_TTP2(TTP,Secret,URL);

claim_TTP3(TTP,Secret,VI);

claim_TTP4(TTP,Secret,Dessision);

Figure 6. The proposed scheme in SPDL-Scyther.

The security of our work is depended on the claim "***no polynomial-time attackers can hack our construction until they reach the following goals:***"

1. Hack the Trusted Third Party (TTP) and access his sensitive data.
2. Restore ID_i from ID_i'.

2) Implementation

To implement and simulate our work, we need to install the XAMPP that supports PHP language software on a computer system containing Windows 10 Enterprise operating system (64 bit), Intel (R) Core (TM) i5-4500U CPU @ 2.70 GHz 2.90 GHz processor, and 4 GB RAM. We used the XAMPP software to simulate the community and authentication servers and PHP, MySQL languages for back-end programming, and HTML5, JS, jQuery, and Bootstrap for front-end programming [20].

2.1. XAMPP Software

XAMPP is a popular cross-platform web server that allows programmers to write and test their code on a local webserver. It was created by Apache Friends, and the audience can amend or modify its native source code. It includes Apache HTTP Server, phpMyAdmin database (MySQL), and interpreters for PHP languages. It's available in 11 languages and runs on a variety of platforms, including Windows, Mac OS, and Linux operating systems[17]. Figure 7 shows the main menu of XAMPP software and Figure 8 explains the XAMPP dashboard.

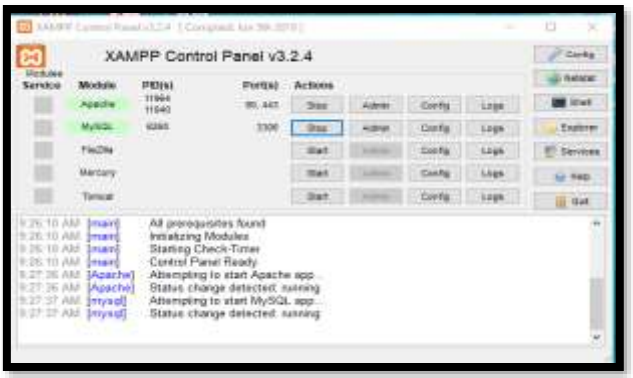


Figure 7. XAMPP Main Menu

Figure 8. The XAMPP's dashboard.

The figure above shows the dashboard of XAMPP software. From this page, we can directly access the database and show the PHP information. This page can be accessed easily by following the below steps:

1. From the main menu window, start Apache and MySQL.
2. Open the browser and type http://localhost/ in the URL's dialog box.

2.2. PHP Language

PHP is a general server-side programming language with a focus on web development. It is an easy, flexible, easy to work programming language[18]. Figure 9 bellows shows the code of the White List (WL) in PHP.

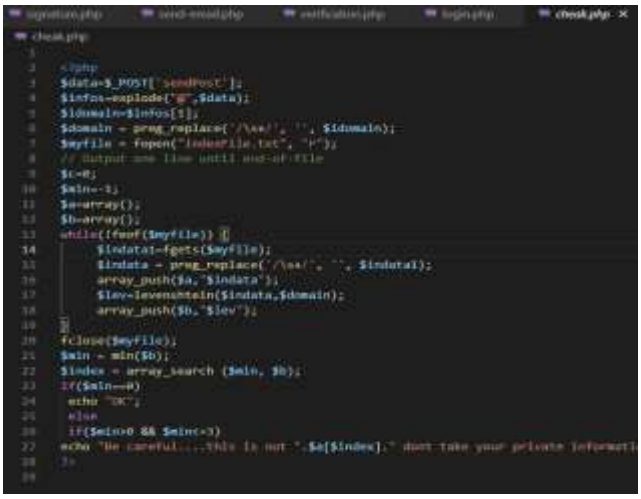


Figure 9. The Code of WL in PHP

2.3. The phpMyAdmin (MySQL)

MySQL is a relational database management system that is free and open-source. MySQL, like other relational databases, organizes data into tables with rows and columns. SQL, or Structured Query Language, is a programming language that allows users to define, manipulate, control, and query data. It's used to store and retrieve data in a wide range of popular apps, websites, and services (using the Apache web server, MySQL database, and PHP for processing). Figure 10 shows the database implementation in the XAMPP software[19].

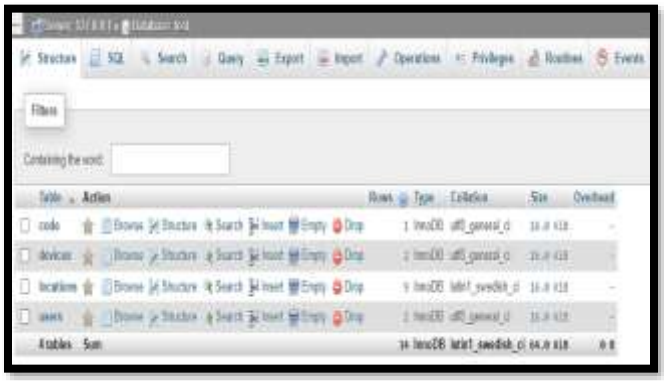


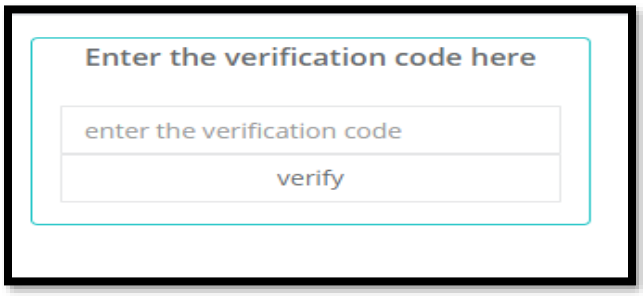
Figure 10. MySQL in XAMPP Software

2.4. The Practical Implementation

In this section, we show our implementation of the proposed scheme using the above-mentioned tools. Figure 11(a, b, c) shows the verification user's interfacing practically.

For the public IP verification property, when the user enters his identity from an unauthorized location, *AS* sends an email to the user containing the verification link as Figure 12. The TTP sets the timestamp value 90 seconds for the validity of the verification link, if the user exceeds this timestamp, the link will be expired.

a. The dialog box for entering the identity



b. The user's dialog box for entering the verification code

c. The verification completion

Figure 11. The user's interfacing for the verification phase

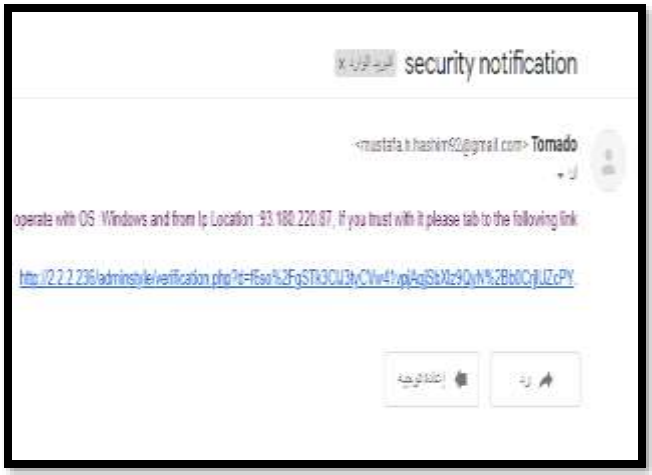


Figure 12. The received email from an authentication server

2.5. Performance Analysis

2.5.1. Communication Cost

We used the following assumptions to compute the cost of sent messages in the verification phase: identity size is 32 bits, hash value size is 160 bits, schnorr digital signature size is 64

bytes (512 bits), the size of an encrypted message is 60 bytes and ECDSA signature size is 72 bytes (576 bits). Based on Table 8, we also compare our proposed scheme to other related works depended on table 3 below. Figure 13 explains the graph of the above comparison.

Table 3. Performance-Based on Communication Messages

Authors	No of bits	No of messages
Our Scheme	**1152**	**5**
Alzuwaini et al. [14]	**1312**	**5**
Bojjagani et al.[7]	**1440**	**5**
Lin et al.[11]	**1536**	**4**
Roy et al.[9]	**864**	**2**

From the above table, we notice that our proposed scheme needs 1152 bits within 5 messages in the verification phase. *Roy et al.*[9] scheme needs 864 bits within 2 messages but it suffers from phishing attack and needs a security analysis

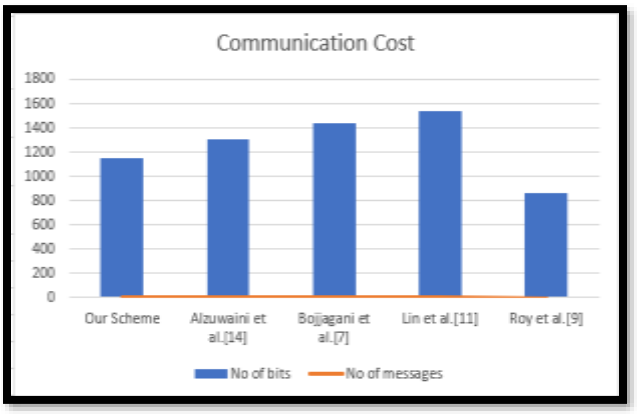


Figure 13 The graph of the Communication Bits comparison.

V. Conclusion

In this paper, we present a secure verification scheme that avoids famous cyber-attacks like phishing, MITM, impersonate, DoS/DDoS, device stolen, and replay attacks based on a symmetric encryption mechanism that offers a high level of security and can detect well-known attacks. Additionally, the proposed scheme has proved using formal security analysis using the scyther tool to prove the exchanged information between legitimate entities are secure and safe against well-known cyberattacks

Our work can be employed in sensitive institutes such as banks, hospitals, and anyone that needs a high level of security. At the time the *TTP* got compromised by an attacker, the attacker will know nothing about the user's information because the user's identity stores in an anomaly form.

References

[1] E. Rostami, F. Karlsson, and S. Gao, "Requirements for computerized tools to design information security policies", *Computers & Security*, vol. 99, no.1, pp. 1-17, 2020 .

[2] L. Ma, et al, "Mitigation of malicious attacks on structural balance of signed networks", *Physica A: Statistical Mechanics and its Applications*, vol. 548, p. 123841, 2020.

[3] A. Vishwanath, "Mobile device affordance: Explicating how smartphones influence the outcome of phishing attacks", *Computers in Human Behavior*, vol. 63, pp. 198-207, 2016

[4] D. Goel, and A. Jain, "Mobile phishing attacks and defence mechanisms: State of art and open research challenges", *Computers & Security*, vol. 73, pp. 519-544, 2018.

[5] J. Rastenis, et al, "E-mail-Based Phishing Attack Taxonomy", *Applied Sciences*, vol. 10, no. 7, p. 2363, 2020.

[6] C. Huang, S. Ma, W.-L Yeh, C. -Y Lin, and C. -T Lee, "Mitigate web phishing using site signatures." In *TENCON 2010-2010 IEEE Region 10 Conference*, pp. 803-808. IEEE, 2010.

[7] S. Bojjagani, D. Brabin, and P. Rao, "PhishPreventer: A Secure Authentication Protocol for Prevention of Phishing Attacks in Mobile Environment with Formal Verification", *Procedia Computer Science*, vol. 171, pp. 1110-1119, 2020.

[8] D. Johnson, A.Menezes., and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)", *International journal of information security* vol. 1, no. 1, pp. 36-63, 2001.

[9] S.Roy, et al, "On the design of provably secure lightweight remote user authentication scheme for mobile cloud computing services", *IEEE Access* vol. 5, pp. 25808-25825, 2017.

[10] A. Ahmed-N, and M. Samovar, "Strong authentication for mobile cloud computing", In *2016 13th International Conference on New Technologies for Distributed Systems (NOTERE)*, pp. 1-6, 2016.

[11] H. Lin, "Efficient mobile dynamic ID authentication and key agreement scheme without trusted servers", *International Journal of Communication Systems* vol. 30, no. 1, p. e2818, 2017.

[12] B. Ross, et al, "Stronger Password Authentication Using Browser Extensions", In *USENIX Security Symposium*, pp. 17-32. 2005.

[13] E. Munivel, and A. Kannammal. "New authentication scheme to secure against the phishing attack in the mobile cloud computing", *Security and Communication Networks* vol. 2019, pp. 1-19, 2019.

[14] Mustafa H. Alzuwaini, and Ali A. Yassin, "An Efficient Mechanism to Prevent the Phishing Attacks", Iraqi Journal for Electrical & Electronic Engineering, Vol. 17, Issue 1, pp. 125-135, 2021.

[15] O. Okunoye, N. Azeez, and F. Ilurimi, "A Web enabled Anti-phishing solution using enhanced Heuristic based technique", vol. 13, no. 2, pp. 304-321, 2017.

[16] C. Cremers, "The Scyther Tool: Verification, falsification, and analysis of security protocols", In *International conference on computer aided verification*, pp. 414-418. Springer, Berlin, Heidelberg, 2008.

[17] D. Dvorski, "Installing, configuring, and developing with Xampp", 2007.

[18] R. Lerdorf, K. Tatroe, and P. MacIntyre, "Programming Php" , *O'Reilly Media, Inc*, 2006.

[19] A. Sofwan, "Belajar Mysql dengan Phpmyadmin", *Universitas Budi Luhur* 2007.

[20] I. Khazal and M. Hussain, "Server Side Method to Detect and Prevent Stored XSS Attack", *Iraqi Journal for Electrical and Electronic Engineering*, Vol. 17, No. 2, pp. 125-136, 2021.