doi:10.2498/cit.1001983

A One-Time Server-Specific Password Authentication Scheme

Adebukola Onashoga[1], Adesina Sodiya[1] and A. Afolorunso[2]

[1] Department of Computer Science, University of Agriculture, Abeokuta, Nigeria
[2] School of Science and Technology, National Open University of Nigeria

Over the years, Password-based Authentication (PA) techniques have been the widely used security mechanism that serves as a first level defence against unauthorised access. However, it is paramount that existing PA techniques should be improved upon in order to adequately protect computer systems and networks from password attacks. This work presents a One-Time Server-Specific Password Authentication Scheme (OTSSPAS) for preventing password related attacks. In this work, two protocols known as Password Joggling Protocol (PJP) and Account Management Protocol (AMP) were developed and integrated with OTSSPAS. PJP involves the use of a Password Security Key (PSK) in order to dissuade adversaries from tapping the password. AMP provides an enhanced account management system by considering previous key activities of users in making account locking decision. OTSSPAS adopts MD5 standard hashing technique for protection of passwords before transmit and storage. Microsoft Visual C# and ASP.Net programming languages were used to implement the design. The evaluation result truly shows that the scheme can prevent common password related attacks.

Keywords: authentication, password joggling, account management, attacks, hashing

1. Introduction

Computer system and networks, which store valuable resources of organisations, are increasingly subject to attacks. The attackers have continually used different techniques to gain access to computer systems. Over the years, many security mechanisms such as identification and authentication, audit trail, firewall, intrusion detection system, encryption etc, have been put in place to prevent unauthorised access resources on computer systems. Consequently, there is a need to continuously improve on current security mechanisms in order to adequately protect computer systems. Most of the security mechanisms that were developed in the past have not completely prevented attacks on computer system. Sodiya et al. (2006) stated that it is necessary to continue to seek ways of improving security of systems so as to be able to efficiently prevent unauthorised access to these systems.

The most widely used security mechanism is authentication. Authentication has two components; namely identification and authorisation. Identification is the means by which a user provides a claimed identity to a system, while authentication is the means of establishing the validity of the claim (NIST, 2002).

The process of authentication requires that the user supplies one or more authentication elements as proof of identity. The combination of identifiers (i.e. credentials) is validated against an authentication database. If the identity is recognised, the user is allowed to have access into the system. Identification and authentication are first line of defence against attacks for most computer systems. Once the system is able to identify and differentiate users, it can enforce access control by preventing unauthorised people or processes from consuming resources and provide user accountability for authorised users (NIST, 2002).

In the past, many forms of authentication were developed. Some of them are password, identification card, external security device and biometric authentication. Biometric authentication uses the unique characteristics of human beings to authenticate their identities. Fingerprints, handprints, faces, retinas, voices, handwriting signatures and keystroke timing can all be tied to a unique individual. The most widely used authentication method is the password. The password is the secret form of authentication data used to control access to resource (Scott-Chapman, 2006).

However, password-based authentication is the most easily subverted method of authentication. If a password is generated from a user's identity, it might be easily guessed and therefore easily cracked. If a security personnel enforces a policy in which elaborate and incomprehensible codes are used, these codes, when written down, are exposed to unauthorised user. Therefore, some of the existing password policies are still weak with low entropy. Other problems of password are safety and proper management of password database.

In this paper, a One-Time-Server-Specific Password protocol that requires that a user remembers a single password consisting of 8 random characters and prevents the 13 prominent password related attacks is proposed. The rest of this paper is organized as follows. In Section 2, some related works and their weaknesses were discussed. The proposed scheme is presented in Section 3. In Section 4, the implementation procedure is described with some display of screenshots captured during the implementation. Evaluation procedure is discussed in Section 5. Finally, conclusion and future work are presented in Section 6.

2. Related Works

In the past, a lot of meaningful contributions were made to the improvement of password authentication (PA) technique. Unfortunately, most of these attempts are relatively inadequate and cannot completely protect computer systems.

Victor and Robert (2004) provided countermeasure for six different PA related attacks. The identified attacks are sniffing attacks, ID spoofing attacks, brute-force attacks, relay attack, and credential decryption. The measures provided are not completely effective.

Blake et al. (2002) mentioned another important attack known as keyboard monitoring. It was shown that there are some daemon applications that can listen to keyboard events sent to the password field and record those keys in some hidden fields. As a result, the daemon obtains the clear text password. This attack made many PA system less effective, it was not considered while developing the many systems.

Viega (2004), Herzog (2001) and Anderson (2001) have made tremendous contributions in identifying PA related attacks. Generally, none of them was able to identity all PA attacks and provide convincing countermeasures. It should be noted that a single and small security hole can cause unimaginable security bridge.

Robert and Sawma (2003) presented countermeasures for 7 e-commerce authentication attacks. The work identified and implemented at least a technique for preventing all the known attacks. Out of the 7 e-commerce authentication attacks presented, the six that are related to password-based authentication are sniffing attacks, ID spoofy attacks, brute-force attacks, Dictionary attacks and credential decryption.

A study of password and methods used in brute-force SSH attacks was presented in Owens and Mathew (2008). In their work, data were collected from a large number of SSH brute-force attacks against linux system connected to different kinds of networks. Patterns in password used these attacks and methods were identified. The work showed that the previous countermeasures, especially for bruteforce attacks are not sufficient.

Lamport (1981) introduced the first hash-based one-time password authentication scheme to defeat wire tapping or sniffing, but his scheme involves very high hash overhead and practical difficulty. Lin et al. (2001) presented an Optimal Strong Password Authentication (OSPA) which claimed to be secure against stolen verifier attack, denial-of-service attack, replay attack and impersonation attack. However, it has been shown by Ku et al. (2002) that OSPA is still vulnerable to stolen verifier attack.

Goyal et al. (2005) presented a scheme to counter online dictionary attack. Their scheme was divided into two protocols. They claimed that the second protocol is more secure than the first protocol. However, their scheme is susceptible to malicious server attack. An attacker who can launch a malicious server attack successfully would not consider exploiting an online dictionary attack.

Mangipudi and Katti (2006) presented a Hash-based Strong Password Authentication Protocol with User Anonymity for better protection of users' passwords. They claimed that the protocol could prevent denial-of-service attack, offline and online guessing attack, stolen verifier attack and replay attack. However, the claim that their work ensures user anonymity and resists stolen verifier attack is wrong. An attacker who has monitored the communication between the user and the server may be able to track the credential information transferred from the former to the latter. The attacker may now obtain the key that can be used to derive the future verifiers from the captured credential information by XORing the appopriate values.

Mohammed et al. (2007) presented an Anti-phishing Single Password Protocol (SPP) which uses a one time password approach in solving sniffing attack and wire tapping, but they have not considered the fact that the server itself represents vulnerability. An adversary, who gains access to the password file on the server, can explicitly gain access to all password verification information, the usernames and the stored random challenge, which greatly increases the chances for a successful dictionary attack. Moreso, the authors have not considered a situation wherein an attacker intercepts a user's login credential information in transit and prevents it from getting to the server.

Kim and Koc (2008) recently presented an improved authentication scheme. Their scheme consists of 4 protocols, namely the Registration protocol, the Login protocol, the "Forget password" protocol and the Password/Verifier-change protocol. The weakness of their scheme is described in the next paragraph.

The authors agreed that an attacker may be able to derive PV in step 1 of the Registration protocol and R_s in step 2 of the login phase, after monitoring the communication between U and S. The attacker can also obtain

$$L^* = h(h(K_u^*||P^*||U_{id}^*) \oplus PV') \oplus h(K_u^*||P^*||U_{id}^*)$$

after replacing the password P with P^*, the user identity, U_{id} with U_{id}^*, but cannot use it to login on the server as the server can detect L as modified. However, an attacker needs not change the values of P, U_{id} and K_u to get authenticated. An attacker simply needs to capture PV in step 1 of the Registration protocol and L at step 3 of the login phase. Albeit the value of L changes at every login due to the dynamic nature of R_s, an attacker may still deduce $h(h(K_u||P||U_{id}) \oplus PV') \oplus h(K_u||P||U_{id}) \oplus PV'$. In order to do this, the attacker needs to capture L at step 3 of the login protocol and derive

$$h(h(K_u||P||U_{id}) \oplus PV') \oplus h(K_u||P||U_{id}) \oplus PV'$$

offline by XORing L with a large range of random nonce, r_s until $R_s = r_s$. Since the attacker now has PV and $h(h(K_u||P||U_{id}) \oplus PV') \oplus h(K_u||P||U_{id}) \oplus PV'$, he may replay PV to S at login and obtain a random nonce, R since PV is static in nature. The attacker thereafter computes

$$h(h(K_u||P||U_{id}) \oplus PV') \oplus h(K_u||P||U_{id}) \oplus PV' \oplus R$$

and use it to login at S successfully.

3. One-Time Server-Specific Password Authentication Scheme (OTSSPAS)

OTSSPAS provides a scheme that improves on Kim and Koc (2008) and other protocols to form an enhanced password authentication.

The major design considerations in OTSSPAS are;

- Improving existing account management systems, specifically the account locking feature, thus preventing denial-of-service attack.
- Provision of an efficient countermeasure strategy for protection against 13 prominent password-related attacks.
- Use of a one-time password resilient against message replay attack.

General Notations in OTSSPAS

- P_u denotes the text containing the PSK entered by the user at the password field.
- P_i denotes the password of the user, ID is the username of a user and PSK is any arbitrary sequence of characters chosen by the user at login.
- i is the length of the user's password $(i = 8)$ and j is the length of the PSK. $(j = 4)$

- U denotes the user's score based on the category of that user.
- h denotes a cryptographic hash function such that $h(a||b)$ implies the hash of a concatenated a and b and $h^2(a) = h(h(a))$.
- $\oplus$ denotes a bit-wise XOR operator.
- n refers to the total number of login attempts.
- E_s denotes the encryption with the public key of the server.
- D_s denotes the decryption with the private key of the server.
- N is a 24-bit random number.
- Alice is a client.
- S' is the domain name of the malicious server.
- S is the domain name of the benign server.

3.1. Password Joggling Protocol

This protocol involves the use of a Password Security Key (PSK) in order to dissuade adversaries from tapping the password from the text entered by the user via key-logging devices. Each password supplied by a user at login is in the form P_u such that;

$$P_u = P_i + \text{PSK}$$

Users are meant to supply their passwords and usernames at registration, but at login, users are expected to supply their passwords along with the password security keys at the password field. The PSK is any arbitrary sequence of characters other than the password characters chosen by the users at login. The password length is chosen to be 8 and the PSK's length is chosen to be 4 so as to confuse a keylogger considerably.

3.2. Account Management

This scheme is introduced in order to prevent account locking from genuine users. A genuine user who mistypes or forgets his password is expected to recollect it only after a few login attempts. Account locking shall be done strictly depending on the integrity of a user. The integrity of a user shall be determined with respect to three factors, namely: the user's time of operation, the user's lock history and the user's category. Each of these factors shall be discussed in turn and a score between 0 and 1 shall be dedicated for attributes within each of these three factors and then users will be awarded with these scores for the attributes they exhibit.

3.2.1. Time of Operation

This factor is aimed at detecting suspicious activities such as online dictionary attacks. It considers the login interval for consecutive login attempts. The time of operation of a user is considered normal if the login interval for three login attempts is not below 3 seconds and odd if vice versa i.e.

$$T_o = \{(0,1) : \mathit{Normal} \equiv 1\,\mathit{point}, \\ \mathit{Odd} \equiv 0\,\mathit{point}\}.$$

3.2.2. Lock History

A user's lock history refers to the number of times a user account has been locked. A user without any history of account lock scores a point of 1 or 0 if vice versa i.e.

$$L_h = \{(0,1) : \mathit{False} \equiv 1\mathit{point}, \mathit{True} \equiv 0\mathit{point}\}$$

3.2.3. Users' Category

There are 3 categories of users in OTSSPAS, namely: the frequent user, the normal user and the rare user. A user shall be categorized based on the total number of successful login attempts denoted as Log, pertaining to that user and the average of the total number of successful login attempts denoted 'A' pertaining to all users. Hence, we have:

$$U = \{(0,0.5,1) : \mathit{Frequent\ user} \equiv 1\,\mathit{point}, \\ \mathit{Normal\ user} \equiv 0.5,\ \mathit{Rare\ user} \equiv 0\,\mathit{point}\}$$

Such that

$$U = 0 \text{ if Log} >= 0.5A \\ U = 0.5 \text{ if } 0.5A < \text{Log} < A \\ U = 1 \text{ if Log} < 0.5A$$

The integrity_Score $= \sum_{n=1}^{\infty}(U + T_o + L_h)/3$.

A user with an integrity score greater than or equal to 0.83 is considered as a user with a high integrity while users with integrity score less than 0.83 are considered as users with low integrity. Account locking is hereby restricted to users with low integrity only after 3 consecutively failed login attempts.

3.3. One-Time-Server-Specific Password Scheme (OTSSP)

The Password Joggling protocol and the Account Management Scheme are now integrated into the One-Time-Server-Specific Password Scheme. The OTSSP scheme is divided into 2 phases, namely: the Registration phase and the Login phase. The two phases are described below.

Registration phase

1. User → Server:
   $h(ID_h||S) \oplus ID_h$, $h(P||ID||S) \oplus ID_h$
   The user inputs his ID and P_u into the client system. The client system separates the user's PSK from P_u to get P_i. Thereafter, the client system computes the user's Temporary Verifier, TV $= h(ID||S) \oplus ID_h$ and the Password Verifier, PV $= h(P||ID||S) \oplus ID_h$ where $ID_h = h(ID)$. Thereafter, the client system sends TV and PV to the server. The server now initializes T_o, L_h and U to a value of 1 and calculates the integrity_score of the user. Then the server stores TV, PV, T_o, L_h, U and the integrity_score of the user in its password file.

Login phase

1. User → Server: $E_s(ID_h, h^2(ID_h||S))$
   The user enters his ID and password into the client system. The client system computes ID_h and $h^2(ID_h||S)$ and encrypts them with the server's public key before sending them to the server.
2. User ← Server: N
   The server derives ID_h and $h^2(ID_h||S)$ by decrypting $D_s(E_s(ID_h, h^2(ID_h||S)))$. Then it verifies $h^2(ID_h||S)$ by XORing the ID_h with the TV stored in the registration phase to obtain $h(ID_h||S)$ and susbequently $h^2(ID_h||S)$. If it verifies the received $h^2(ID_h||S)$ as being correct, it sends a random challenge, N to the client system.
3. User → Server: P_R
   The client system now computes the one-time Message Authentication Code (MAC) denoted as
   $$\begin{aligned} P_R = & h^2(P_i||ID||S) \oplus ID_h \\ & \oplus h^2(ID_h||S)||h^2(h(P||ID||S) \oplus N) \end{aligned}$$
   and sends it to the server. The server starts to verify P_R against the TV and PV stored in the registration phase. Thus, the server derives $h(ID_h||S)$ and $h(P_i||ID||S)$ by respectively XORing the TV and PV stored in the registration phase with the ID_h derived in step 2 of the login phase. The server now computes $h^2(P_i||ID||S) \oplus ID_h \oplus h^2(ID_h||S)||h^2(h(P||ID||S) \oplus N)$ and checks if it matches with P_R received from the user before granting a successful authentication.
4. User ← Server: Success/Fail
   The server notifies the user of either a successful or a failed authentication attempt. The server updates T_o, L_h and U and the integrity_score of the user. In the event of a failed login attempt, the server checks the integrity_score of the user as well as the value of n, before making decisions about locking the user's account.

Assumptions

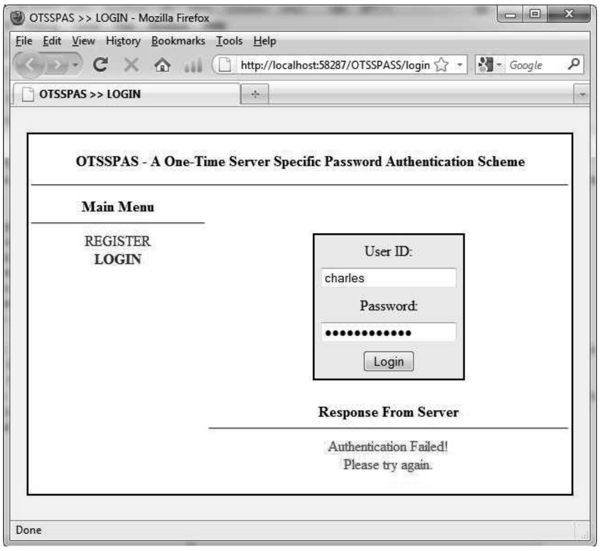
There are 2 assumptions in OTSSP Authentication Scheme (OTSSPAS). The first is that before OTSSPAS can execute the Secure Socket Layer (SSL), SSL must have authenticated the server using PKI certificate mechanism so that a malicious server may not spoof another server's domain name successfully. The second assumption is that users may provide valid telephone numbers or e-mail addresses during registration so as to provide users who may find their accounts locked in the event of authentication with a way to unlock their accounts. To this effect, links or URL will be provided to users via e-mails to direct them to a non-spoofed interface within the web page on which they had registered their accounts in order to change their passwords before reactivating their accounts. Temporary authentication codes may be sent to their mobile phones for reactivating their accounts when they find difficulty in accessing their accounts.

4. Implementation and Security Analysis

This section discusses the implementation procedure and different ways of analysing the protocol considering common attacks.

4.1. Implementation

OTSSPAS was implemented with Visual C# on ASP.Net platform. The following screenshots display some of the modules in the proposed protocol. Figure 1 illustrates the response from the server indicating a failed login attempt by a user after entering an incorrect password.


OTSSPAS >> LOGIN - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:58287/OTSSPASS/login
Google
OTSSPAS >> LOGIN
Done

OTSSPAS - A One-Time Server Specific Password Authentication Scheme

Main Menu

REGISTER
LOGIN

User ID: charles

Password: ••••••••••••

Login

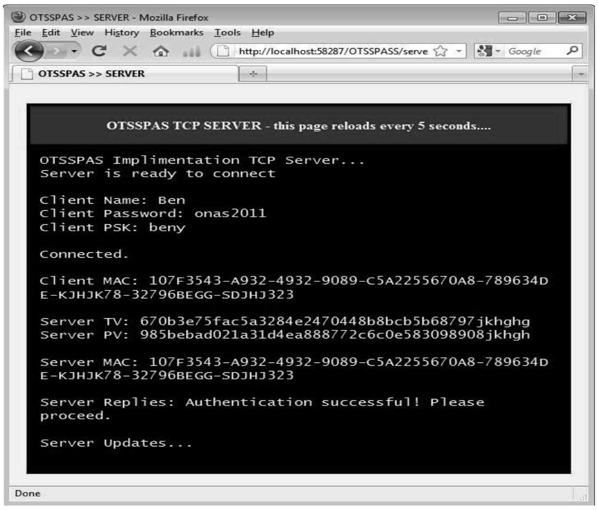
Response From Server

Authentication Failed!
Please try again.

Figure 1. The login phase of OTSSPAS.

The server, implemented with ASP.Net is designed to autoload/refresh every 5 seconds to reflect the log saved in the database. The server's response to one of the login processes is shown on the screenshot displayed in Figure 2.

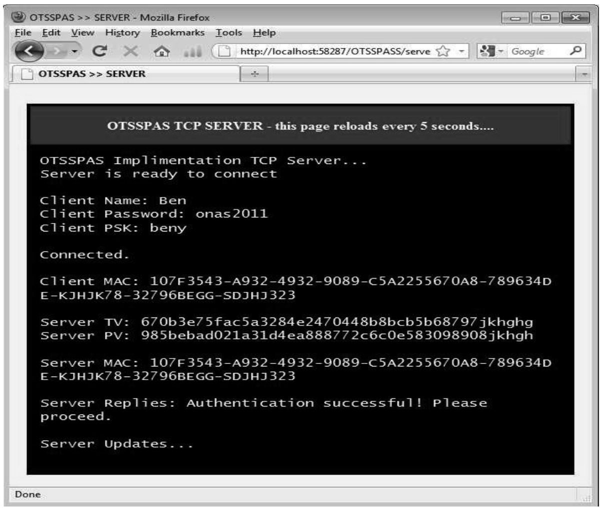
Figure 3 shows a detailed information about the integration of the PJP and AMP. It displays the Client name, Client Password and PSK. The Client computes the MAC after separating the actual password from PSK. The server computes MAC using the stored TV and PV. On carrying out this computation, there is an equality in the Client MAC and Server MAC as seen on the screenshot. The Client is then authenticated and server updates the components of the AMP.


OTSSPAS >> SERVER - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:58287/OTSSPASS/serve
Google
OTSSPAS >> SERVER
Done

OTSSPAS TCP SERVER - this page reloads every 5 seconds....

```
OTSSPAS Implimentation TCP Server...
Server is ready to connect

Client Name: Ben
Client Password: onas2011
Client PSK: beny

Connected.

Client MAC: 107F3543-A932-4932-9089-C5A2255670A8-789634D
E-KJHJK78-32796BEGG-SDJHJ323

Server TV: 670b3e75fac5a3284e2470448b8bcb5b68797jkhghg
Server PV: 985bebad021a31d4ea888772c6c0e583098908jkhgh

Server MAC: 107F3543-A932-4932-9089-C5A2255670A8-789634D
E-KJHJK78-32796BEGG-SDJHJ323

Server Replies: Authentication successful! Please
proceed.

Server Updates...
```

Figure 2. Server response for incorrect login.


OTSSPAS >> SERVER - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:58287/OTSSPASS/serve
Google
OTSSPAS >> SERVER
Done

OTSSPAS TCP SERVER - this page reloads every 5 seconds....

```
OTSSPAS Implimentation TCP Server...
Server is ready to connect

Client Name: Ben
Client Password: onas2011
Client PSK: beny

Connected.

Client MAC: 107F3543-A932-4932-9089-C5A2255670A8-789634D
E-KJHJK78-32796BEGG-SDJHJ323

Server TV: 670b3e75fac5a3284e2470448b8bcb5b68797jkhghg
Server PV: 985bebad021a31d4ea888772c6c0e583098908jkhgh

Server MAC: 107F3543-A932-4932-9089-C5A2255670A8-789634D
E-KJHJK78-32796BEGG-SDJHJ323

Server Replies: Authentication successful! Please
proceed.

Server Updates...
```

Figure 3. Server response for correct login.

4.2. Security Analysis

An analysis of OTSSPAS was conducted on a LAN within the Computer Science Lab in the University of Agriculture, Abeokuta, Nigeria. 100 registered students with the department's site within this network were given access to the systems within a period of 6 weeks in order to evaluate the security of OTSSPAS against each of the considered password-related attacks.

This section briefly demonstrates that our proposed scheme is secure against some attacks, to mention but a few, keylogging attack, dictionary attack, brute-force attack and server spoofing attack.

4.2.1. Keylogging attack

A keylogging attack is aimed at tapping the user's password after tracking or logging the keys struck on the keyboard so that the retrieved password may be used in an online dictionary attack. A keylogging software was installed on each of the systems in the Computer Science Lab in order to monitor the login of each student. After the students had accessed the computer systems over a period of 6 weeks, it was observed that 25% of the students chose passwords that can be found inside the English dictionary, 70% of the students chose passwords that are names combined with other characters, 1.5% of the students chose passwords that could not be ascribed to any meaning while 3.5% of the students chose passwords that are combination of characters and numbers. A list of the passwords entered at the password field by the students was compiled. The captured list of passwords was tested at a later time in order to login at the server after the account locking feature of OTSSPAS had been disabled. None of the collected passwords were used to login successfully. This was as a result of the inability to locate the actual password within the sequence of characters entered into the password field by the students at login.

4.2.2. Dictionary attack

An attacker tries to login with every word in a dictionary in this kind of attack. However, OTSSPAS is secure against this kind of attack because the integrity of such attacker pretending to be a user continues to diminish until the attacker is locked out.

A total number of 5 attack dictionaries were collected from the internet with an aim of deducing if the passwords chosen by the students could be found on the lists. The passwords listed in the dictionary contained words typically with length of less than 8. None of the passwords chosen by the students were found in the attack dictionaries.

An attacker tries every word in a password dictionary against a password verification information in order to derive the password. However, taking the average number of guesses in an offline dictionary attack to be a standard value of 10000000 and the standard time required in hash computation to be 0.005 ms, an attacker needs up to $(68^8 \cdot 2^{24} \cdot 10\,000\,000 \cdot 0.005 \cdot 0.001)$ seconds to derive the user's password. It is thus practically infeasible to derive the user's password in an offline dictionary attack.

4.2.3. Brute-force attack

In order to test the system with brute-force attack, thirty more students that had not partaken in the previous exercises were supplied with the password file of the students who had earlier registered with the server. The 30 students had a job to supply password guesses to the system until they were able to derive any MAC found in the password file. This exercise lasted for a total of 14 hours within 4 different days. Yet, the students could not derive any MAC listed in the password file.

An attacker launches this kind of attack by first obtaining the password file containing the password verification information of all users and for every possible combination of characters, the attacker tests whether it is the correct password. OTSSPAS is secure against this type of attack because the attacker requires testing up to a total of 68^8 combinations of characters alongside 2^{24} different combinations of 24-bit random number simultaneously, which takes years to accomplish.

4.2.4. Server spoofing attack

A malicious server aims at stealing the challenge stored by the benign server for a client, such that that client also has an account with the malicious server. The malicious server thus replays the same challenge stolen from the benign server to its clients with the aim of receiving the MAC that it may use to login at the benign server at a later time. OTSSPAS is secure against this kind of attack because each MAC is specific to the server that the MAC is sent to. An attacker cannot forge another server's identity because SSL handles server authentication using PKI certificates. A simple illustration of a scenario wherein an attacker obtains the MAC from the user after sending him/her a stolen challenge is given below.

1. Alice → Server: $E_s(ID_h, h^2(ID_h||S'))$
2. Alice ← Server: N
3. Alice → Server: $h^2(P_i||ID||S') \oplus ID_h \oplus h^2(ID_h||S'))||h^2(h(P||ID||S') \oplus N)$
   It is to be noted that $P_R = h^2(P_i||ID||S) \oplus ID_h \oplus h^2(ID_h||S)||h^2(h(P||ID||S) \oplus N)$ is different from what is sent from Alice to the server at step 3. Hence a malicious server cannot replay what it receives from Alice back to the benign server.

5. Conclusion

OTSSPAS successfully addresses the 13 prominent password-related attacks. A password file compromise on the server will only result in leaking the temporary User Verifiers, UV and the Password Verifier, PV, but not the Message Authentication Code (MAC) which is actually used in verifying both verifiers. The use of the MAC has also provided a One-Time Server Specific Password Scheme for OTSSPAS because the MAC used in identifying each user is computed dynamically with respect to the challenge the user receives from the server as well as the server's identity. Hence, OTSSPAS is clearly resilient to message replay attack, and malicious server attack. The server is involved with only a total of 4 computations of a one-way forward hash function in order to authenticate each user in OTSSPAS. This shows that the computational load on the server is less, which in turn enhances the efficiency of OTSSPAS. Finally, the evaluation results show that OTSSPAS can prevent 13 related password attacks.

References

[1] R. Anderson, *Security Engineering: A Guide to Building System*, John Wiley and Sons, 2001, ISBN: 0-471-389226-6.

[2] Anti-phishing Working Group (2007), *Phishing activity trend: Report for the month of November 2007*, `http://www.antiphishing.org/reports/apwg_report_nov_2007.pdf`

[3] R. Blake, J. Collin and M. Nick, *Stronger Password Authentication: using Browser Extentions*, 2002.

[4] M. Charles, *Authentication System: Password and Passphare*, 2005, Retrieved from `http://www.xtermin.ud`.

[5] V. Goyal, V. Kumar, M. Singh, A. Abraham & S. Sanyal, A new protocol to counter online dictionary attack, *Journal of Computer and Security*, 2006, 25(2), pp. 114–120.

[6] P. Herzog, *The Open Source Security Testing Methodology Manual*, version 1.5, 2001, Retrieved from `http://www.ideahanster.org`

[7] B. Ives, K. Walsh, and H. Scneider, The Domino Effect of Password Reuse, *Communication of the ACM*, 2004, 47(4), pp. 75–78.

[8] M. Kim and C. K. Koc, A Secure Hash-Based Strong-Password Authentication Protocol Using One-Time Public-Key Cryptography, *Journal of Information Science and Engineering*, 2008, 24(4), pp. 1213–1227.

[9] W. Ku, H. Tsai and S. Chen, Two simple attacks on Lin-Shen-Hwang's strong-password authentication protocol, *ACM SIGOPS Operating Systems Review*, 2003, Vol. 37, No. 4, pp. 26–31.

[10] W. Ku, A hash-based strong-password authentication scheme without using smart cards, *ACM SIGOPS Operating Systems Review*, 2004, Vol. 38, No. 1, pp. 29–34.

[11] L. Lamport, Password Authentication with Insecure Communication, *Communications of the ACM 1981*, 24:770–2.

[12] C. Lin, J. Shen and M. Hwang, Security Enhancement for Optimal Strong-Password Authentication Protocol, *ACM SIGOPS Operating Systems Review*, 2003, Vol. 37, No. 2, pp. 7–12.

[13] Mangipudi and Katti, A Hash-based Strong Password Authentication Protocol with User Anonymity, *International Journal of Network Security*, 2006, Vol. 2, No. 3, pp. 205–209.

[14] G. Mohammed, M. Alex, M. Lok & A. Mohamed, An anti-phishing Single Password Protocol, *Journal of Computer and Telecommunications Networking*, 2007, 51(4), pp. 3715–3726.

[15] NIST, National Institute of Standard and Technology (2001), *Underlying Technical Models for information Technology Security*, Special Publication 800-33 Retrived from `http://www.csrc.nist.gov/publications`

[16] J. Owens and J. Mathew, A Study of Password and Method used in Brute-Force SSH Attacks, 2008, `http://people.clarkson.edu/owensjp/pobs`

[17] B. Pinkas and S. Thomas, Securing Passwords Against Dictionary Attacks, in *Proceedings of the 9th ACM conference on computer and communications security 200*, 2003, pp. 161–170.

[18] R. Probert and V. Sowma, E-commerce Security: A New Methodology for Deriving Effective Countermeasures Design Models, in *Proceedings of the 16th Annual Federal Information System Security Educator's Association (FISSEA) Conference*, 2003, http://csrc.nist.gov/organisations/fisseal

[19] A. Scott-Chapman, *Perimeter based, Community-Centric Access Control System*, M. sc. Thesis, Florida State University, 2006.

[20] S. Seungjae, C. Jerry, R. Jungwoo and E. T. Jack, A study of two-factor authentication against on-line identity theft, in *Proceedings of the 39th Annual Meeting of Decision Science Institute, DSI 2008*.

[21] A. S. Sodiya, S. A. Onashoga and O. B. Ajayi, Towards Building Secure Software Systems, Issues in *Informing Science and Information Technology*, Vol. 3, 2006.

[22] D. Victor and L. Robert, E-commerce Authentication: An Effective Countermeasures Design Model, in *Proceedings of the 5th International Conference on Enterprise Information Systems*, 2003, pp. 447–455, 2004.

[23] J. Viega and G. McGraw, *Building Secure Software*, Addison-Wesley, 2002.

[24] Wikipedia (2010), *Random Password Generator*, Free Encyclopedia

[25] The CAPTCHA Project:
http://www.captcha.net/

Received: May, 2011
Revised: February, 2012
Accepted: March, 2012

Contact addresses:

Adebukola Onashoga
Department of Computer Science
University of Agriculture
Abeokuta, Nigeria
e-mail: bookyy2k@yahoo.com

Adesina Sodiya
Department of Computer Science
University of Agriculture
Abeokuta, Nigeria
e-mail: sinaronke@yahoo.co.uk

A. Afolorunso
School of Science and Technology
National Open University of Nigeria
e-mail: releafolorunso@yahoo.com

Adebukola Onashoga is a lecturer at the Department of Computer Science, University of Agriculture, Abeokuta, Nigeria. She has a Ph.D. degree in computer science, with focus on computer security. She has published in both international and local journals. She has attended and presented papers at reputable international conferences. Her current research interests include computer security, data mining and artificial intelligence.

Adesina Sodiya is a senior lecturer and postgraduate coordinator at the Department of Computer Science, University of Agriculture, Abeokuta, Ogun State, Nigeria. He is the present Chairman of the Publications, Standards, Research and Development Committee of Nigeria Computer Society. His current research interests include computer security, software engineering, artificial intelligence and information management. He has published about 24 journal articles in both local and international publication. He also has ten conference proceedings. He is the present Editor-in-Chief of Journal of Computer Science and Its Application.

A. Afolorunso is a lecturer at the School of Science and Technology, National Open University of Nigeria. She is the Programme Leader for Information and Communication Technology. She has attended local and international conferences. She is currently on her PhD programme in computer science at University of Lagos, Nigeria. She has published in some journals. Her research areas are computer security and network systems.