Hindawi
Security and Communication Networks
Volume 2018, Article ID 6315039, 10 pages
https://doi.org/10.1155/2018/6315039

*Research Article*

# Assessment of Secure OpenID-Based DAAA Protocol for Avoiding Session Hijacking in Web Applications

**Muhammad Bilal**,[1,2] **Muhammad Asif**,[2] **and Abid Bashir**[2]

[1]*College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China*
[2]*Department of Computer Science, National Textile University, Faisalabad 38000, Pakistan*

Correspondence should be addressed to Muhammad Bilal; mbilal@zju.edu.cn

Received 7 August 2018; Accepted 26 September 2018; Published 1 November 2018

Academic Editor: Vincenzo Conti



It is increasingly difficult to manage the user identities (IDs) of rapidly developing and numerous types of online web-based applications in the present era. An innovative ID management system is required for managing the user IDs. The OpenID lightweight protocol is a better solution to manage the user IDs. In an OpenID communication environment, OpenID URL is not secured in a session hijacking situation because in other existing OpenID communication methods such double factor authentication has more chances of valid user session hijacked. The proposed communication protocol secures the OpenID URL with the help of additional innovative parameters such as Special Alphanumeric String (SAS) and Special Security PIN (SSP). The anticipated triple authentication protocol authenticated client unique OpenID URL at OpenID Provider (OP) side once and SAS and SSP field at Relying Party (RP) side. The anticipated protocol provides unique Single-Sign-On (SSO) services to OpenID users. The experimental website is tested by experts of web developers for avoiding session hijacking situation in the presence of hackers. The findings demonstrated that Dense Authentication Authorization and Accounting (DAAA) protocol minimizes the risk of a session hijacking in OpenID communication environment.

## 1. Introduction

Identity Management is a vital administrative area in the major field of Information Technology (IT) and security. There are many ways to create and manage front and back-end user authentication and authorization through digital identities. The entire world uses the Internet for various purposes. The Internet has become a part of social life in which users have many accounts to manage. Users create duplicate user profiles for accessing numerous types of online login web-based applications in this IT era. To avoid the duplication of profiles, OpenID protocol has provided SSO services to its users. The main feature of SSO is to enhance the user login process. The SSO is a famous distributed protocol and OpenID supports this protocol [1–13].

OpenID is an open standard and decentralized approach [14–18]. Users in OpenID register an identity provider once and then use the OpenID-enabled websites. OpenID allows using an existing account to sign in to multiple websites, without creating a new account. OpenID protocol also removes the duplication of user profiles in multiple websites logins. In OpenID communication environment, OpenID URL is an identity that facilitates SSO services to its users. The OpenID authentication process for end-users uses standard HTTP requests and responses and does not require any special capability of user agents or client software. OpenID provides users with convenience and better security compared to others login systems. OpenID was created in summer 2005 as an open source related community to solve the problem of multiple logins on different websites. This was done by the OpenID Foundation [19–23]. The OpenID protocol is a simple lightweight protocol [24, 25].

OpenID protocol supports many well-known websites such as VeriSign Lab, Yahoo, Google, ClaimID, GetOpenID, Live Journal, Facebook, Microsoft, Flicker, Hyves, Blogger, OrangeTM, Myspace, Word Press, mixi, and AOL for SSO facilities. In all well-known websites, the biggest issues faced are those of signing into multiple websites. The OpenID

Foundation acts as a public trust-based organization that solved these multiple websites' issues representing the global community in June 2007 [20, 26–30].

OpenID protocol consists of three major components:

(i) Relying Party (RP).

(ii) Identity Provider (IdP) or OpenID Provider (OP).

(iii) User Agent (UA) [1, 2, 12, 20, 29–34].

The above-mentioned three major components are basically interconnected with each other for communication in the OpenID-based environment. OpenID protocol's normal working environment is shown in Figure 1 [26].

There are different steps involved in the OpenID protocol working environment as shown in Figure 1.

(i) Clients (UA) request the RP to access the websites by providing the ID.

(ii) RP communicates with the OP for authentication of clients (UA).

(iii) OP redirects to clients (UA) for authentication.

(iv) Authorized Clients (UA) communicate with the OP for authentication.

(v) The OP sends a response to the RP (success or failure for client's authentication).

(vi) The RP provides clients (UA) access to start the communication properly [1, 26, 35].

OpenID communication-based environment provides multiple benefits, such as strong authentication for valid and authentic user access in OpenID related websites that easily facilitate the OpenID-enabled users [36]. The OpenID members avail many benefits such as cost saving and collaboration among multiple industries and users. The client can easily login through OpenID without giving username and password information to the server through a Relying Party or user website. The client can manage the information sent to requesting websites at all risk levels easily. In OpenID, communication environment, username, and password are not being written anywhere with a remember option [26, 36]. OpenID authentication protocol is the most famous among multiple users for using multiple websites, blogs, and social sites in the form of single profiles' creation [25].

There are two modes of communication between the Relying Party (RP) and OpenID Provider (OP):

(i) Dumb Mode

(ii) Smart Mode

In the dumb mode, RP (consumer) does not maintain the state of association between OP and RP and more steps are used for completion of authentication. In Smart mode, RP maintains the state of association between OP and RP and fewer steps are used for authentication [27].

The session hijacking situation occurs when hackers steal the valid user session information for a specific website and act as an authorized user. Session hijacking is just like a man in the middle attack and has significantly higher rates than security attacks [26, 37–39].

The session hijacking is still a threat for web-based applications, that is, significant loss of important data such as banking details and online user login credentials. Most researchers started working for the protection of session hijacking in mid-1990s. The researcher Nikiforakis proposed a session shield technique as protections against session hijacking. The researcher Dacosta team proposed One-Time-Cookies (OTC) solution as protections against session hijacking [40].

Authentication means to verify the legitimacy process and is performed in different forms such as textual, biometric, token, and graphical based password [41].

There are different methods used for authenticating the user depending on the situation:

(i) Password-based authentication (desktop, web-based, and client-server applications)

(ii) PIN-based authentication (ATM, telephone card system)

(iii) Smart card based authentication (scan and plugged into the system)

(iv) Biometric authentication (scan fingerprint and eye retina)

(v) Certificate-based authentication (X.509, SSL certificate) [27, 42]

Authentication of users via OpenID URL in OpenID communication-based web applications is a major issue faced still today due to insecurity and unreliability because of session hijacking risk. How can secure and reliable user authentication in an OpenID communication environment be ensured for online web applications in a session hijack situation?

In this paper, a secure OpenID protocol is designed and implemented in an experimental website to demonstrate protection against session hijacking. In this innovate protocol, OpenID URL is secured by using two parameters SAS and SSP at client / RP side. The experimental website test results exhibited that DAAA protocol is more secure in an OpenID communication-based environment. Section 2 in this paper presented related work carried out by other researchers. Section 3 described the design methodology in detail. Section 4 elaborated proposed protocol implementation with an explanation. In the end, Section 5 presented the conclusion and future work.

## 2. Related Work

The web-based attack is a huge challenge for web applications on both the client and server sides. Session hijacking is one of the techniques in which hackers/intruders steal the session ID of the client/user in the communication server or website and act as an authorized user. In an OpenID communication environment, a hacker is a person that is dangerous for authentic clients because if security is implemented in a simple and normal way then the hacker works as an authorized user between the client and server sides and endangers the user

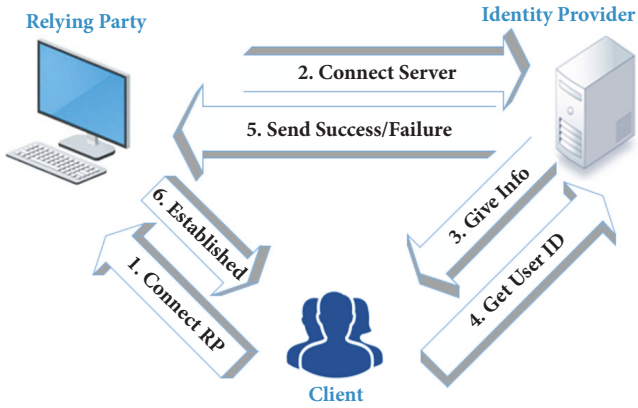


Figure 1: OpenID communication environment [26].

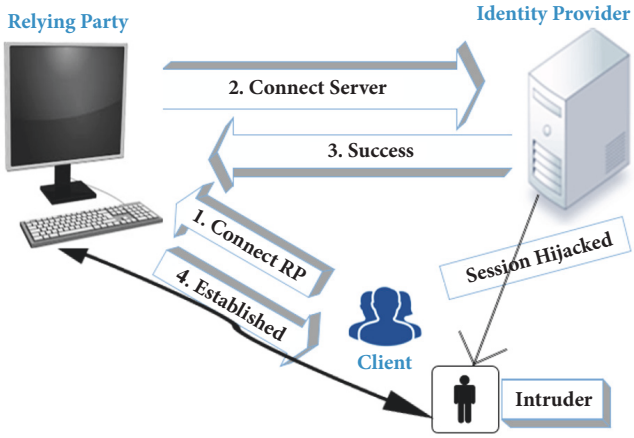


Figure 2: Session hijacking situation in OpenID environment [26].

data. In this situation, the OpenID URL is not secured as shown in Figure 2 [26].

Simply the hacker steals the user identity from the communication server or website as shown in Figure 2. In this paper, to avoid a session attack on the client side, a double authentication communication scheme is used. In this scheme, the PIN code is used as an additional parameter for user authentication. The user information and PIN code as an additional parameter are stored in an identity provider once ready for initiating communication in an OpenID-based environment. Yadis' discovery protocol is used for locating the identity of the OpenID URL in this model. The RP and OP communicate with each other by using the Diffi Hellman algorithm to share an association message. When a user request for RP to communicate is established successfully with the PIN code, then RP redirects the user to the identity provider side through the Yadis protocol to check the OpenID identity. If the OpenID URL identity is valid, then the identity provider server sends a success message with the additional parameter PIN code to RP. After receiving the successful message RP comparison, the user's originally requested PIN and identity are provided back with the response PIN for authentication of the valid user. When both PINs are matched on the RP side then successful communication is established

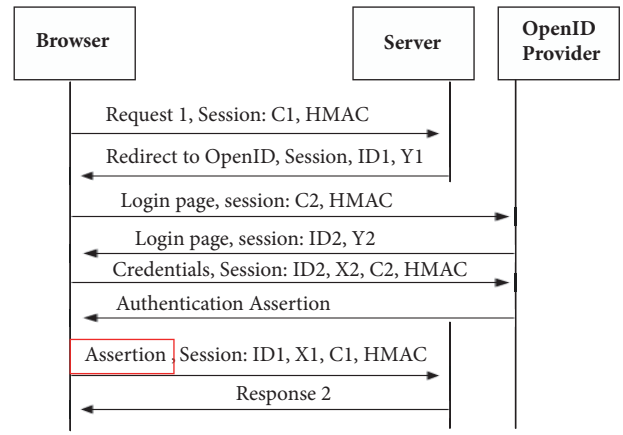


Figure 3: OpenID authentication using secure session management [43].

between the RP and the authentic user. In simple term, a double authentication scheme can be done by using one authentication method on the identity provider side and the second authentication method on the RP side. But if the PIN on the client side is stolen by a hacker then that issue still exists in session hijacking, as the use of PIN is considered as a weak form of authentication [26].

In this paper, to avoid phishing attacks on the server side, two-factor based antiphishing schemes are used in the OpenID environment. The two additional credentials of the password and Personal Identification Number (PIN) are used as double factor authentication in this model. The two additional credentials have been tested on the experimental website to avoid phishing attacks against theft. This prototype system is tested by using pip VeriSign Labs as an identity provider [1].

This paper describes session management as an important part of the user's authentication in current web-based applications at the server side. The session is secured effectively via the HyperText Transfer Protocol (HTTP) channel in web applications by sharing the secret key between the browser and the server side. The signature request idea is used between the browser and the server. If the session is prevented by attackers, the secret key is not accessible or transmitted due to the third-party authentication in the form of a valid signature request checked. The valid signature is signed and verified by hash-based message authentication code (HMAC). This secret key is also used to secure the OpenID authentication scenario as shown in Figure 3 [43].

Figure 3 represents how the OpenID-based authentication works in secure session management by using third-party authentication. The important observation in Figure 3 is an assertion trigger that is marked by an oval symbol. This assertion is used by the Relying Party as a claimed identity for verifying the session authentication. If attackers try to hack the session, then ID1 (Session Identifier) is used to solve the problem by binding each assertion of the session identifier. The de facto solution is implanted for securing the Transport Layer Security (TLS) [43].

Researchers in the past have already successfully developed methods to reduce the risk of attacks through the password-based solution, for example, ATM PIN number. But password-based applications user faced problems in case of forgetting, resetting after some days, the security questions asked. Nowadays, new technology is used in most web applications for securing the password, such as two-factor authentication, biometrics, tokens, and federated identity system such as OpenID. Security experts analyzed the poor security that users faced on multiple web-based attacks such as session hijacking and client-side malware, as proved by US Government National Strategy for Trusted Identities in Cyberspace (NSTIC) in 2011. The researchers analyzed user dependency upon password-based authentication works best if the password is managed by good policies and is cost-effective [42].

The security limitations in an OpenID communication environment for users' online authentication are the malicious Relying Party that redirects the user to wrong or fake websites, considered as a phishing attack, and hackers hijacking the valid user session that is still an issue in this field [3].

In this paper, two-factor BeamAuth web authentication techniques are used for avoiding the phishing attack in the Single-Sign-On (SSO) form. This technique requires bookmarking the real estate and browser setup measures. The BeamAuth solution exploits the URL (used as an identifier) and other unusual properties and cannot send to network even in the case of page changes or page reloading. Secondly, new security features include testing and deploying the web application without client updating information [44].

The rapid growth of Internet services requires a problem to create a new account for accessing these services. But this user login complexity decreased by SSO and OpenID somehow. This paper basically analyzed existing cloud computing techniques strengths and weakness. In this paper, an anticipated model is the one in which One Time Password (OTP), Trusted Platform Module (TPM), and OpenID are used for avoiding phishing attacks in the cloud-based environment [35].

This paper analyzed security issues and weakness of OpenID protocol through the Open Web Application Security Project (OWASP) tools. The high-level new OpenID integration model with Higgins is proposed for enhanced security advantages. In this integration model, all messages of OpenID protocol flow must be SSL/TLS protected. But one of the major limitations of this model cannot protect against dishonest identity providers [30].

The main issues of SSO were faced in form of several attacks today like spam, session hijacking, link manipulation, and phishing. In this paper authors basically solved the issues of phishing attack through page token mechanism. The page token prototype was developed with google console developers and saleforce.com. The page token evaluation effectiveness was successfully done in control laboratory experiments by involving 26 participants of high institutes of Malaysia northern regions. The experiment results were conducted by dividing participants into two groups' page token and without page own token equal basis. The page token performance with page token successfully login valid user against phishing attack as compared to without page token unsuccessfully login against phishing attacks [13].

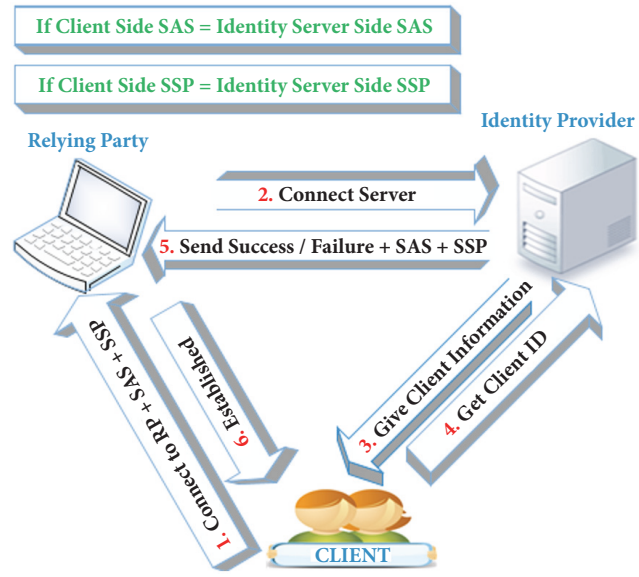


Figure 4: DAAA OpenID authentication protocol (proposed protocol).

## 3. Methodology

The anticipated model is designed with the help of OpenID communication-based components: OpenID Provider (OP), Relying Party (RP), and client agents for securing the OpenID URL identity. The proposed OpenID protocol modifies previous existing relevant research work and enhances user online authentication scenarios in a secure manner through the Dense Authentication Authorization and Accounting (DAAA) protocol. The OpenID DAAA protocol suggests the use of two additional innovative parameters of Special Alphanumeric String (SAS) and Special Security PIN (SSP) for securing the unique OpenID URL of user identity scenario as shown in Figure 4.

Figure 4 represents a secure DAAA proposed protocol in OpenID communication-based environment. There are several necessary steps involved in the secure OpenID protocol communication environment as depicted in Figure 4. In this protocol, OpenID URL user identity is secured with the help of two innovative parameters, SAS and SSP.

In the SAS parameter, field client login in the OpenID-based communication environment is performed on the client side via 4 alphanumeric characters. The 4 alphanumeric characters are provided by the client in such a manner that the 1st number is numerical form (any 0...9), the 2nd number is capital letter form (any A...Z), the 3rd number is small letters form (any a...z), and the 4th number is a special character from the most commonly used keys of the keyboard [any ! @ # $ % * ( ) / | + - ) ]. The above-mentioned 4-characters are used in any sequence, for example, "B1b@".

In the SSP parameter, field restricted 4-digit numbers are provided similarly as a PIN (any 0. . ..9 four-digit numbers), for example, "5240".

According to the proposed design strategy, the client registers on the identity provider side and each client gets a unique identity in the form of OpenID URL. The first step is client interaction with RP, with two innovative parameters: SAS and SSP. The second step is RP redirection of the client to the identity provider side with the help of the Yadis discovery protocol. The third step is the identity provider asking the client to send the user ID information. The fourth step is the client sending the user ID information to the identity server. The fifth step after verifying the user identity at the identity server side is sending a successful response to RP with two innovative parameters SAS and SSP. If the identity is not matched, then send a failure response to the RP side. The sixth step is comparing the SAS and SSP parameter fields of the client at the RP side with the back response of the identity provider side SAS and SSP at RP side. If both of the two authentications are compared successfully at the RP side, then authorized communication is established between the RP and the client agent side, otherwise a failure. Simply, in this DAAA OpenID protocol, the security features of user login through OpenID URL are enhanced by innovative modification of Diffie-Hellman key exchange algorithm.

Triple authentication means first the authentication of identity in the form of OpenID URL at the identity server side and another two authentications SAS and SSP at the client side. These are done correctly to avoid a session hijacking situation in the OpenID-based communication environment.

## 4. Implementation

The proposed protocol is developed in the powerful core PHP application. According to the planned design, no well-known identity providers provide a facility for any researcher experiment itself. Due to this main security drawback, the protocol itself designed an identity provider and client-side application to conduct an experiment. In this developed strategy, the client and the RP side act the same. First, register the client at identity server side and get a unique identity in the form of an OpenID URL. The registration form consists of important user details such as username, password, SAS, SSP, full name, email, mobile no, address, city, and country fields. After signup, each client gets the unique identity in the form of an OpenID URL. After getting a unique OpenID URL identity, the client cannot change the unique identity but password, SAS, and SSP field can be changed if the client requires login with username and password in the OpenID Provider portal side. After getting an OpenID URL, the client interacts with the client-side/RP side portal through the OpenID URL identity.

Figure 5 depicts the client-side application interface. In this interface, the registered client interacts with the OpenID RP side through an OpenID URL identity. When a client provides the OpenID URL and logins, then the client-side redirects to the identity server side with the help of the Yadis discovery protocol. If an OpenID URL is correctly provided by the user, then the OpenID Provider verifies the password of the registered client, otherwise, an invalid URL and password message will be displayed in the client-side application. In simple words, the first authentication of the client can be done at the identity provider side if an OpenID URL and password is provided correctly and redirects clients to another interface of the client-side portal with two additional parameters, SAS and SSP, for 2nd and 3rd authentication. If any provided OpenID URL or password is wrong, then the failure message is displayed in the client-side portal and will not redirect to another client-side interface.

Figure 6 illustrates the OpenID client app side in which the valid OpenID URL shows the form after a successful 1st authentication of the client at the OpenID Provider side. In the form, the second field represents the SAS field and third represents SSP field. These two fields are used for 2nd and 3rd authentication at the client-side portal. When the OpenID client fills out the form field of SAS and SSP and logins, then the OpenID Provider response of two additional form fields of SAS, SSP, and RP side are compared. If both SAS and SSP fields of the RP side portal match with the server response SAS and SSP then the RP side two authentications complete successfully. The valid user is then redirected to an authorized user interface; otherwise, they cannot be redirected to an authorized user portal.

Figure 7 signifies the authorized client portal. After successful triple authentication then authorized user accesses this portal.

*4.1. Programming Code.* The important code for avoiding session hijacking at RP/client side by using two additional parameters' (SAS and SSP) field (triple authentication) in an innovative way as shown in Algorithm 1.

The session code in Algorithm 1 is managed in an innovative way at RP side for avoiding session hijacking situation in OpenID communication environment innovative way. In this code 2nd and 3rd authentication of the valid user are done by comparing the response of identity server-side response of SAS and SSP parameter field.

The flow of the DAAA OpenID-based experimental website is shown in Figure 8.

The experiment results were conducted by some well-known experienced professional web developers in the IT field by using the nonprobability technique of judgment sampling. In the designed protocol guide, the important points of using the system were explained. The following important points are as follows:

(i) Getting a unique identity (sign up form information)

(ii) SAS (4 alphanumeric characters how to use)

(iii) SSP (4 digits number how to use)

(iv) Triple authentication scheme for avoiding session hijacking

The mentioned above four steps are very much important for implementation of secure DAAA OpenID communication protocol.

To get more insight in this manuscript, the sample (consisting of 63) was collected through the judgment sampling

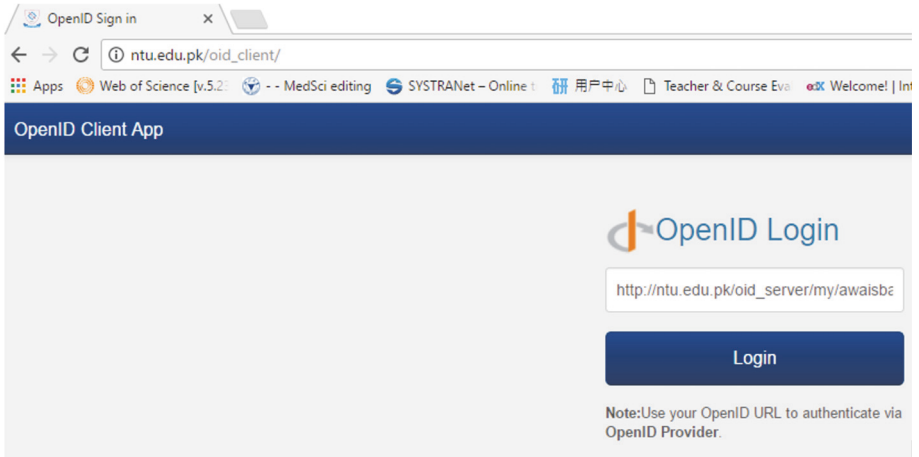


Figure 5: Client/RP side portal for 1st authentication.

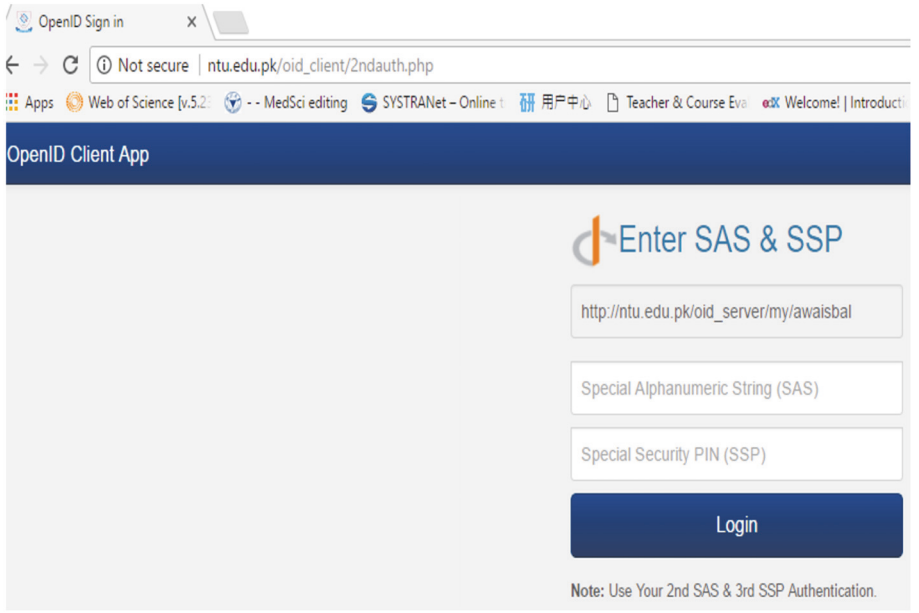


Figure 6: Client/RP side portal for 2nd and 3rd authentication.

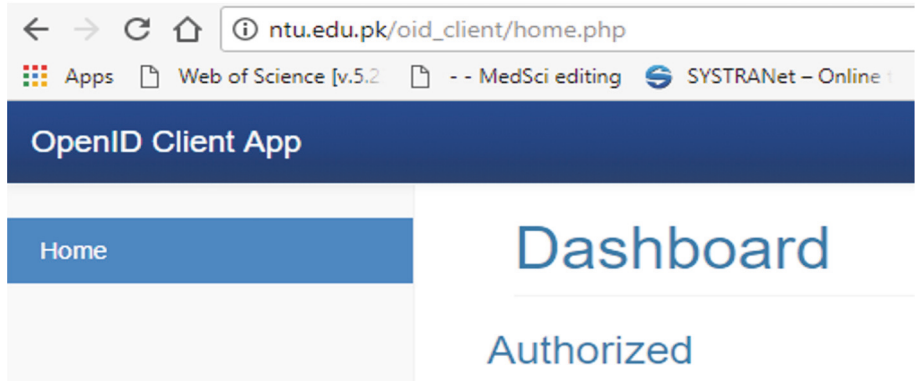


Figure 7: Authorized client portal.

Table 1: Assessment of the proposed protocol model with existing OpenID model.

| OpenID Methods | Login any PC/Laptop (Any/Own) | Security (Safe/ Unsafe) | Price | Trust |
|---|---|---|---|---|
| Preventing Session Hijacking [26] | Any (Verisign labs Personal Identity Portal used for the experiment) | Partially Safe (Double Authentication) | Free | Medium |
| Anti-phishing OpenID Solution [1] | Any | Partially Safe (Double Authentication) | Free | Medium |
| BEAMAUTH Anti-phishing solution [44] | Own (vanilla web browser) | Partially Safe (Double Authentication) | Low-cost | Low |
| OpenID Authentication Model using Trusted Computing [35] | Any | Partially safe | Low-cost | High |
| OpenID security weakness analyzed via OWASP tools [30] | Own (web browser) | Not very much Secure | Free | Low |
| Anti-phishing OpenID Page token solution [13] | Own (google console developers and salesforec.com) | Partially secure | Low-cost | Medium |
| **Proposed DAAA Protocol** | **Any** | **Safe (Triple Authentication)** | **Free** | **High** |

∗Bold observations provide best security features as compared to other OpenID methods.

scheme from the professional web developers as shown in Figure 9. The following five concerning questions were asked of these professionals to check the security aspects of the new experimental website after use:

(i) Is the experimental website easily understandable for users?

(ii) Is the experimental website providing better security in session hijacking situations for valid user authentication?

(iii) Is the experimental website reliable in session hijacking situation for valid user authentication?

(iv) Is the experimental website provide a better idea of using multiple websites to secure a Single Sign-On (SSO) method?

(v) Has the experimental website secured the OpenID URL in an OpenID communication environment?

According to the new experimental website (82.53%) respondents approved that the experimental website is easily understandable. The experimental website provides better security in session hijacking situations (90.06%). The majority of the respondents (93.65%) felt that the experimental website is more reliable in session hijacking situations. Approximately 84.12% of the respondents preferred that the experimental website provides a better idea of using multiple websites in a secure SSO way. The 82.53% of respondents felt that an OpenID URL is secure in the OpenID-based communication environment. Figure 8 also demonstrated support of numerical results in terms of all the above-asked questions. Therefore, it is also concluded both numerically and graphically that the experimental website is more secure. The anticipated method is also compared with already existing OpenID communication methods.

The summarized analysis of OpenID communication environment with different methods is shown in Table 1. In triple authentication, scheme hacker has less chance to hijack the valid user session because security scheme is implemented in an innovative and decent way. Also, the hacker has less chance to hijack the client-side valid user authentication scenario because two innovative parameters of SAS and SSP authentication can be done at the client/ RP side.

The summarized analysis of OpenID communication environment different methods is shown in Table 1.

Table 1 examined that the proposed model is better than the existing OpenID methods with all three aspects of login, security, and expenses point of view.

## 5. Conclusion and Future Work

The proposed solution is more secure in session hijacking situation in OpenID communication-based environment by comparing already existing OpenID URL security methods as mentioned in Table 1. It secures the OpenID URL of the valid user by using two innovative additional parameters, Special Alphanumeric String (SAS) and Special Security PIN (SSP). Therefore, these two additional parameters of SAS and SSP in the proposed DAAA protocol cannot be hijacked in an OpenID communication environment. It is

```
session_start();
//error_reporting(E_PARSE);
$error = "0";
// Default space so that height of page does not shrink
if(isset($_REQUEST["userurl"]) &&
isset($_REQUEST["sas"]) && isset($_REQUEST["ssp"])
&& isset($_REQUEST["checkauth"]) &&
isset($_REQUEST["token"])) {
        if ($_SESSION["reverse_token"] ==
$_REQUEST["token"] && $_REQUEST["checkauth"] ==
"1") {
        //Put Values in Session
                $_SESSION["token"] =
$_REQUEST["token"];
                $_SESSION["SAS"] =
$_REQUEST["sas"];
                $_SESSION["SSP"] =
$_REQUEST["ssp"];
                $_SESSION["userURL"] =
$_REQUEST["userurl"];
        }
}
elseif(isset($_REQUEST["userurl"]) &&
isset($_REQUEST["usas"]) && isset($_REQUEST["ussp"])
&& isset($_REQUEST["login"]) &&
isset($_REQUEST["token"])) {
        if ($_SESSION["reverse_token"] ==
$_REQUEST["token"] && $_REQUEST["sas_ssp"] == "1")
{
                if ($_REQUEST["usas"] !=
$_SESSION["SAS"]) {
                        $error = "Invalid 2nd SAS
Authentication";
                }
                elseif ($_REQUEST["ussp"] !=
$_SESSION["SSP"]) {
                        $error = "Invalid 3rd SSP
Authentication";
                }
                else {
                unset($_SESSION["SAS"]);
                unset($_SESSION["SSP"]);
                $_SESSION["isLoggedIn_oid"]=1;
        // Session Variable
                        $_SESSION["sessid_oid"] =
session_id();
                        header("location: home.php");
                }
        }
    }
```

Algorithm 1

concluded that the session cannot be hijacked in the DAAA protocol for OpenID URL. It has been also tested via an experimental website. Hackers will have less possibility of hijacking the valid user session, and the proposed system is secure and reliable after successful login through the triple authentication method.

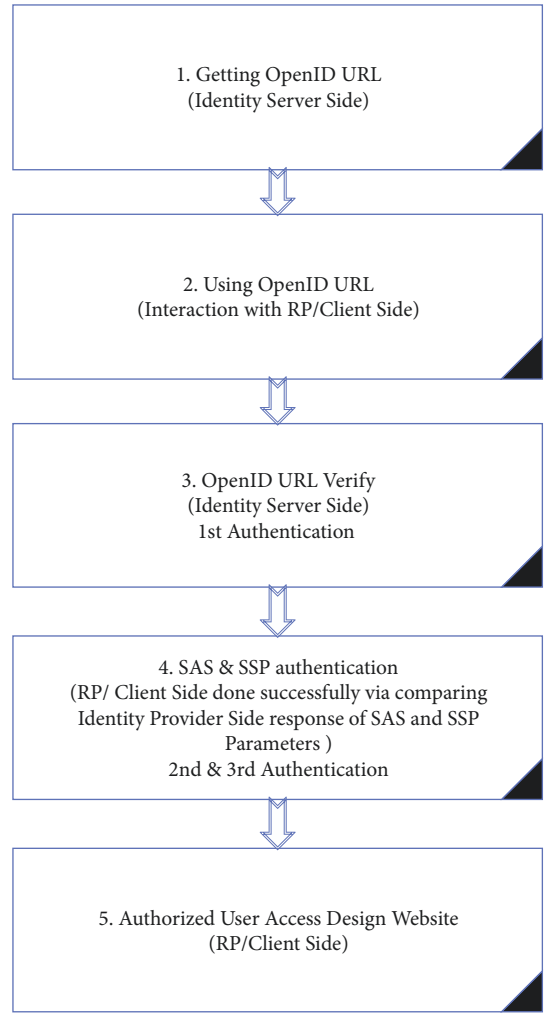


Figure 8: Flow of DAAA OpenID protocol in experimental website.

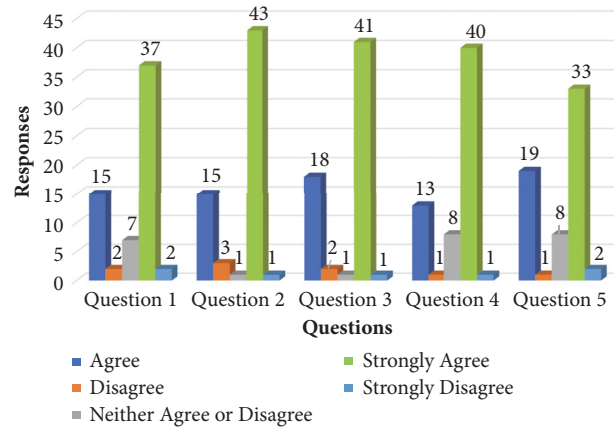


Figure 9: Dense AAA OpenID protocol experimental website results.

The OpenID communication environment will be made more secure in session hijacking situations with the help of image recognition such as the face, eye contacts, or thumb expression. But this is an expensive solution in the client's point of view. The SSO technique for instance OpenID will become more and more important in future client point of view.

## Data Availability

References used for related work of this paper within the text, figures, and tables are available. All data are available freely (online, journal papers, and conference articles).

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

The authors gratefully acknowledge Mr. Joseph Bass, University of North Florida (UNF), and Mr. Imran Memon, Zhejiang University, China, for some useful discussions and editing of this paper. Special acknowledgement is to all Zhejiang University teachers for their support in this paper. This work is supported by Alibaba-Zhejiang University Joint Institute of Frontier Technologies, Zhejiang Provincial Natural Science Foundation of China (Grant no. LZ13F020001), National Science Foundation of China (Grants nos. 61173185 and 61173186), and National Key Technology R&D Program (Grant no. 2012BAI34B01).

## References

[1] M. Asif, M. S. Sarfraz, M. S. Ahmed, and N. Tripathi, "A Two Factor Based Anti-Phishing Method in Open ID," *Journal of Basic and Applied Scientific Research*, vol. 3, pp. 26–33, 2013.

[2] S.-T. Sun, K. Hawkey, and K. Beznosov, "Systematically breaking and fixing OpenID security: Formal analysis, semi-automated empirical evaluation, and practical countermeasures," *Computers & Security*, vol. 31, no. 4, pp. 465–483, 2012.

[3] H. Oh and S. Jin, "The Security Limitations of SSO in OpenID," in *Proceedings of the 2008 10th International Conference on Advanced Communication Technology*, pp. 1608–1611, Gangwon-Do, South Korea, Feburary 2008.

[4] R. Rehman, *Get ready for OpenID*, 2008, Get ready for OpenID.

[5] S. Mukhopadhyay and D. Argles, "An anti-phishing mechanism for single sign-on based on QR-code," in *Proceedings of the International Conference on Information Society, i-Society 2011*, pp. 505–508, UK, June 2011.

[6] V. Mladenov, C. Mainka, and J. Schwenk, *On the security of modern Single Sign-On Protocols: Second-Order Vulnerabilities in OpenID Connect*, 2015.

[7] D. Fett, R. Kusters, and G. Schmitz, "The Web SSO Standard OpenID Connect: In-depth Formal Security Analysis and Security Guidelines," in *Proceedings of the 2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pp. 189–202, Santa Barbara, CA, USA, August 2017.

[8] A. Leicher, A. U. Schmidt, and Y. Shah, "Smart OpenID: a smart card based OpenID protocol," *SEC*, pp. 75–86, 2012.

[9] B. van Delft and M. Oostdijk, "A Security Analysis of OpenID," in *Policies and Research in Identity Management*, vol. 343 of *IFIP Advances in Information and Communication Technology*, pp. 73–84, Springer Berlin Heidelberg, Berlin, Heidelberg, Germany, 2010.

[10] Q. Feng, K.-K. Tseng, J.-S. Pan, P. Cheng, and C. Chen, "New anti-phishing method with two types of passwords in OpenID system," in *Proceedings of the 5th International Conference on Genetic and Evolutionary Computing (ICGEC '11)*, pp. 69–72, Xiamen, China, September 2011.

[11] C. Mainka, V. Mladenov, and J. Schwenk, "Do not trust me: Using malicious IdPs for analyzing and attacking single sign-on," in *Proceedings of the 1st IEEE European Symposium on Security and Privacy, EURO S and P 2016*, pp. 321–336, Germany, March 2016.

[12] J. Bellamy-McIntyre, C. Luterroth, and G. Weber, "OpenID and the enterprise: A model-based analysis of single sign-on authentication," in *Proceedings of the 15th IEEE International Enterprise Distributed Object Computing Conference (EDOC), EDOC 2011*, pp. 129–138, Finland, September 2011.

[13] N. H. Zakaria, M. F. Zainul, N. Katuk, H. M. Tahir, and M. N. Omar, "An Evaluation of Page Token in OpenID Single Sign on (SSO) to Thwart Phishing Attack," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 10, pp. 19–23, 2018.

[14] D. Recordon and B. Fitzpatrick, *Network Working Group Internet-Draft*, 2007.

[15] P. Sovis, F. Kohlar, and J. Schwenk, "Security analysis of OpenID," in *Proceedings of the Sicherheit 2010 - Sicherheit, Schutz und Zuverlassigkeit Beitrage der 5. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft fur Informatik e.V. (GI) - 5th Annual Conference of the Department of Security of the Society for Informatics*, pp. 329–340, Germany, October 2010.

[16] S. Feld and N. Pohlmann, "Security analysis of openid, followed by a reference implementation of an nPA-based OpenID Provider," in *Proceedings of the 12th Information Security Solutions Europe Conference, ISSE 2010*, pp. 13–25, Germany, October 2010.

[17] R. H. Khan, J. Ylitalo, and A. S. Ahmed, "OpenID authentication as a service in OpenStack," in *Proceedings of the 2011 7th International Conference on Information Assurance and Security, IAS 2011*, pp. 372–377, Malaysia, December 2011.

[18] A. Armando, R. Carbone, L. Compagna, J. Cuéllar, G. Pellegrino, and A. Sorniotti, "An authentication flaw in browser-based Single Sign-On protocols: Impact and remediations," *Computers & Security*, vol. 33, pp. 41–58, 2013.

[19] J. Fishburn, *What is OpenID?*, 2016, http://openid.net/get-an-openid/what-is-openid.

[20] F. Alecu, P. Pocatilu, G. Stoica, C. Ciurea, and S. Capisizu, "OpenID, a Single Sign-On Solution for E-learning Applications," *Journal of Mobile, Embedded and Distributed Systems*, vol. 3, pp. 136–141, 2011.

[21] A. Lindholm, *Security evaluation of the OpenID protocol*, Skolan för datavetenskap och kommunikation, Kungliga Tekniska högskolan, 2009.

[22] J. Krolo, M. Šilić, and S. Srbljić, "Security of Web level user identity management," in *Proceedings of the 32nd International Convention Proceedings: Digital Economy - 6th ALADIN, Information Systems Security, Business Intelligence Systems, Local Government and Student Papers*, pp. 93–98, Croatia, May 2009.

[23] E. Tsyrklevich and V. Tsyrklevich, *Single sign-on for the Internet: a security story*, vol. 340, 2007.

[24] A. Leicher, A. U. Schmidt, and Y. Shah, "Smart OpenID: a smart card based OpenID protocol," *IFIP International Information Security Conference*, pp. 75–86, 2012.

[25] D. Choukse, U. K. Singh, D. Sukheja, and R. Shahapurkar, *Implementing new-age authentication techniques using openid for security automation*, 2010.

[26] A. Muhammad and N. Tripathi, "Evaluation of OpenID-Based double-factor authentication for preventing session hijacking in web applications," *Journal of Computers (Finland)*, vol. 7, no. 11, pp. 2623–2628, 2012.

[27] R. U. Rehman, *The OpenID Book-A comprehensive guide to OpenID protocol and running OpenID enabled web sites*, Conformix Technologies Inc, 2008.

[28] J. Fishburn, *OpenID Connect FAQ and Q&As*, 2016, https://openid.net/connect/faq/.

[29] M. Uruena and C. Busquiel, "Analysis of a privacy vulnerability in the openid authentication protocol," *IEEE Multimedia Communications, Services and Security*, 2010.

[30] W. A. Alrodhan and A. I. Alqarni, "Security Investigation and Analysis of OpenID: Problems and Enhancements," *International Journal of Computer Science and Network Security*, vol. 17, pp. 198–211, 2017.

[31] D. Hardt, J. Bufu, and J. Hoyt, "OpenID attribute exchange 1.0-final," vol. 5, p. 11, 2007.

[32] V. Radha and D. H. Reddy, "A Survey on Single Sign-On Techniques," *Procedia Technology*, vol. 4, pp. 134–139, 2012.

[33] P. J. Riesch and X. Du, "Audit based privacy preservation for the OpenID authentication protocol," in *Proceedings of the 2012 12th IEEE International Conference on Technologies for Homeland Security, HST 2012*, pp. 348–352, USA, November 2012.

[34] F. Alaca and P. C. van Oorschot, "Comparative Analysis and Framework Evaluating Web Single Sign-On Systems," 2018.

[35] E. Ghazizadeh, Z. S. Shams Dolatabadi, R. Khaleghparast, M. Zamani, A. A. Manaf, and M. S. Abdullah, "Secure OpenID Authentication Model by Using Trusted Computing," *Abstract and Applied Analysis*, vol. 2014, Article ID 561487, 15 pages, 2014.

[36] J. Fishburn, *Membership Benefits*, 2016, http://openid.net/foundation/benefits-members/.

[37] N. Nikiforakis, W. Meert, Y. Younan, M. Johns, and W. Joosen, "SessionShield: Lightweight Protection against Session Hijacking," in *Engineering Secure Software and Systems*, vol. 6542 of *Lecture Notes in Computer Science*, pp. 87–100, Springer Berlin Heidelberg, Berlin, Heidelberg, Germany, 2011.

[38] J. Louis, *Detection of session hijacking*, 2011.

[39] D. Madhavi and V. R. Siddhartha, "A Survey on Detection Tools and Prevention Techniques for Session Hijacking Attack," *International Journal of Scientific Research Development (IJSRD)*, vol. 2, pp. 63–66, 2015.

[40] V. Jain, D. R. Sahu, and D. S. Tomar, "Session Hijacking: Threat Analysis and Countermeasures," in *Proceedings of the in International Conference on Futuristic Trends in Computational Analysis and Knowledge Management*, 2015.

[41] A. Ahmad, M. Asif, M. K. Hanif, and R. Talib, "Secure Graphical Password Techniques against Shoulder Surfing and Camera based Attacks," *International Journal of Computer Science and Information Security*, vol. 14, pp. 808–815, 2016.

[42] C. Herley and P. Van Oorschot, "A research agenda acknowledging the persistence of passwords," *IEEE Security & Privacy*, vol. 10, no. 1, pp. 28–36, 2012.

[43] P. De Ryck, L. Desmet, F. Piessens, and W. Joosen, "Improving the security of session management in web applications," 2013.

[44] B. Adida, "Beamauth: Two-factor web authentication with a bookmark," in *Proceedings of the 14th ACM Conference on Computer and Communications Security, CCS'07*, pp. 48–57, USA, November 2007.