An Analysis of Open Standard Identity Protocols in Cloud Computing Security Paradigm

Nitin Naik and Paul Jenkins
Defence School of Communications and Information Systems
Ministry of Defence, United Kingdom
Email: nitin.naik100@mod.uk and paul.jenkins683@mod.uk

***Abstract*—Cloud computing enables businesses to use computing resources on-demand anywhere in the world without having to build and maintain computing infrastructures in-house. This model involves multiple parties performing diverse operations via the Internet across multiple organisations. Employees and consumers can access resources and services from their own and associated organisations. Despite the success of cloud computing, its security paradigm has one major challenge: how to determine the identity and access rights of users across all the organisations. The user's credential and sensitive information are always stored and maintained by the parent organisation, however, other partner organisations require verification of the user's identity and access rights to allow them to access their services and resources. The biggest difficulty is to communicate the user's identity to their partner organisations without sending their sensitive information. Numerous open standard identity protocols have been introduced in the last two decades. Amongst all, three standard identity protocols Security Assertion Markup Language (SAML), Open Authentication (OAuth), and OpenID Connect (OIDC) are the most established protocols in the cloud computing industry. Therefore, this paper presents a working prototype and critical analysis of these three open standard identity protocols SAML, OAuth and OIDC. It also explores evaluation criteria which are used for this analysis purpose. Finally, it discusses their strengths and limitations, and determines the most suitable open standard identity protocol for all types cloud computing models.**

***Keywords*—Cloud Computing Security, Open Standard Identity Protocols, SAML, OAuth, OpenID Connect, SSO, IDaaS**

I. Introduction

The connotation of cloud computing is "computing everywhere" that offers all the computing resources on-demand everywhere via Internet [1]. Cloud computing enables enterprises and consumers to increase their productivity and efficiency of their work by sharing resources and services anytime, anywhere. However, its success comes with a number of security challenges [2], [3]. One of the major challenges is that how to authenticate and authorize employees and consumers across the business organisations. The parent organisation owns all the credentials and sensitive information of users but partner organisations require verification of users' identities and access rights for allowing them to access their services and resources. The biggest problem to overcome is how to communicate users' identities to their partners without sending their sensitive information. The solution to this problem is open standard identity protocols, which are used to authenticate and authorize users across all the business organisations [1], [2].

Open standard identity protocols have been developed to support all authentication and authorization activities at a corporate level. There are various open standard identity protocols available, however, the most popular and successful protocols are SAML, OAuth, and OIDC. SAML is an XML-based protocol framework for communicating user authentication, entitlement and attribute information [4]. OAuth is a scalable delegation protocol that allows a user to grant access to an application to perform authorized tasks on behalf of the user [5]. OpenID Connect is a lightweight protocol that provides a framework for communicating identity via RESTful APIs [6]. These three protocols are used by the majority of cloud computing organisations. Now, identity management services became a stand-alone IT business called IDaaS (IDentity-as-a-Service). Consequently, these open standard identity protocols are crucial for authentication and authorization in cloud computing security paradigm. Therefore, this paper presents the working prototype and critical analysis of these three open standard identity protocols SAML, OAuth and OIDC. In addition, it explores evaluation criteria which are used for this analysis purpose. Finally, it discusses their strengths and limitations, and determines the most suitable open standard identity protocol for all types cloud computing models. The remainder of this paper is organised as follows: Section II elucidates the working prototype of the three open standard identity protocols SAML, OAuth and OIDC; Section III expounds the comparative analysis of these three open standard identity protocols; Section IV critically analyses and determines the most suitable open standard identity protocol for all types of cloud computing; Section V concludes the paper and suggests some future areas of extension.

II. Working Prototype of Open Standard Identity Protocols: SAML, OAuth and OIDC

A. *Security Assertion Markup Language (SAML)*

SAML is an XML-based protocol framework for communicating user authentication, entitlement and attribute information [4]. It allows the two federation partners to choose and share desirable identity attributes in a SAML assertion (message) payload as long as those attributes can be represented in XML [6]. SAML presumes three main roles in any transaction: Identity Provider (IDP), Service Provider (SP) and User. A working prototype of SAML is illustrated in Fig. 1. The SAML assertion (security token) is the core concept in SAML. The SAML assertion is a claim, statement, or declaration of an identity that consists of the IDP and trusted by the SP. The IDP and SP normally agree up front what information the SP requires. However, there is a mechanism to renegotiate

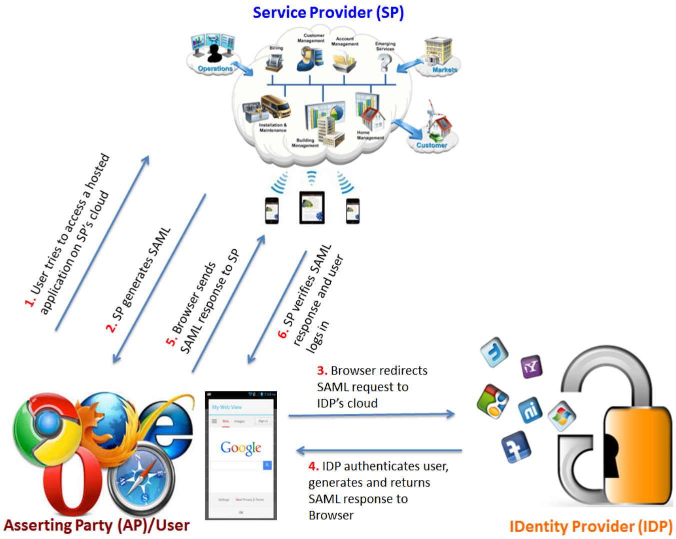
Service Provider (SP)
Operations
Billing
Customer Management
Account Management
Emerging Services
Markets
Installation & Maintenance
Building Management
Home Management
Customer
1. User tries to access a hosted application on SP's cloud
2. SP generates SAML
5. Browser sends SAML response to SP
6. SP verifies SAML response and user logs in
3. Browser redirects SAML request to IDP's cloud
4. IDP authenticates user, generates and returns SAML response to Browser
Asserting Party (AP)/User
IDentity Provider (IDP)

Fig. 1. A working prototype of SAML

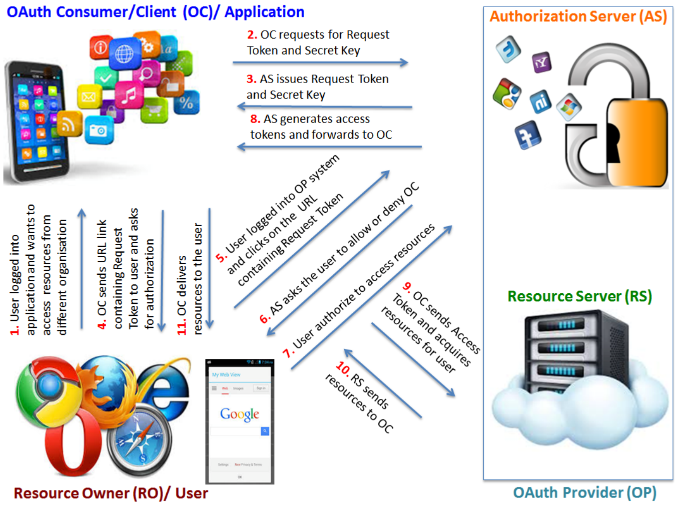
OAuth Consumer/Client (OC)/ Application
Authorization Server (AS)
2. OC requests for Request Token and Secret Key
3. AS issues Request Token and Secret Key
8. AS generates access tokens and forwards to OC
1. User logged into application and wants to access resources from different organisation
4. OC sends URL link containing Request Token to user and asks for authorization
11. OC delivers resources to the user
5. User logged into OP system and clicks on the URL containing Request Token
6. AS asks the user to allow or deny OC
7. User authorize to access resources
9. OC sends Access Token and acquires resources for user
10. RS sends resources to OC
Resource Server (RS)
Resource Owner (RO)/ User
OAuth Provider (OP)

Fig. 2. A working prototype of OAuth

additional information.

B. Open Authorization (OAuth)

OAuth is a scalable delegation protocol that allows a user to grant access to an application for performing authorized tasks on behalf of the user [5]. Thus, it enables a third-party application to obtain limited access to a HTTP service. This secure API authorization can be performed from desktop, web and mobile applications. It introduces the concept of an authorization token that states the right of the client application to access authorized services on the server. However, it does not supersede any access control decisions that the server-side application might make. The OAuth 2.0 core authorization framework is described by IETF in RFC-6749 alongside with several other specifications and profiles [7]. OAuth presumes four main roles in any authorization process Resource Server (RS), Resource Owner (RO)/User, OAuth Consumer/Client (OC) and Authorization Server (AS) [8]. A working prototype of OAuth is illustrated in Fig. 2.

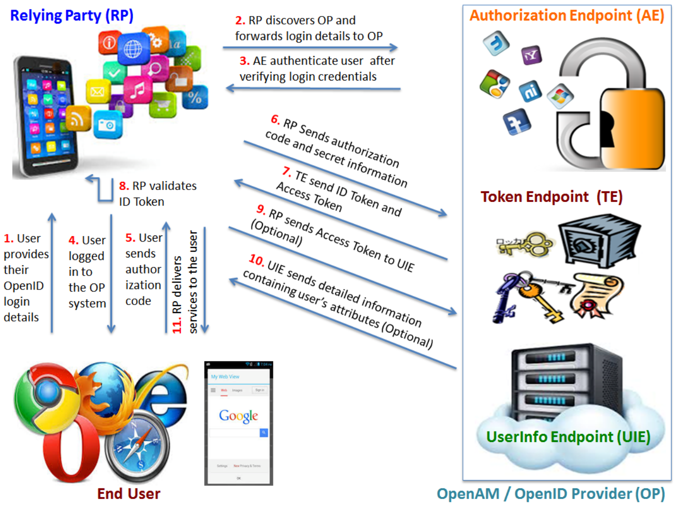
Relying Party (RP)
2. RP discovers OP and forwards login details to OP
3. AE authenticate user after verifying login credentials
Authorization Endpoint (AE)
6. RP Sends authorization code and secret information
7. TE send ID Token and Access Token
9. RP sends Access Token to UIE (Optional)
10. UIE sends detailed information containing user's attributes (Optional)
8. RP validates ID Token
1. User provides their OpenID login details
4. User logged in to the OP system
5. User sends author ization code
11. RP delivers services to the user
Token Endpoint (TE)
UserInfo Endpoint (UIE)
End User
OpenAM / OpenID Provider (OP)

Fig. 3. A working prototype of OIDC

C. OpenID Connect (OIDC)

OIDC is a lightweight protocol that provide a framework for communicating identity via RESTful APIs [6]. OpenID Connect 1.0 is a simple identity layer on the top of the OAuth 2.0 protocol [6], [9]. It enables clients to verify the identity of the end user based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the end user. OpenID Connect facilitates two primary types of tokens: an Access Token and an ID Token. The ID token is a JWT (JSON Web Token) and contains information about the authenticated user. It is signed by the identity provider and can be read and verified without accessing the identity provider [10]. OIDC presumes five main roles in any authentication and authorization process End User (EU), Relying Party (RP), Authorization Endpoint (AE), Token Endpoint (TE) and UserInfo Endpoint (UIE) [10]. A working prototype of OIDC is illustrated in Fig. 3.

III. Comparative Analysis of Open Standard Identity Protocols: SAML, OAuth and OIDC

Table I expounds the comparative analysis of these three open standard identity protocols SAML, OAuth and OIDC on the basis of several explored criteria.

IV. Critical Analysis of Open Standard Identity Protocols: SAML, OAuth and OIDC

The comparative analysis has demonstrated the characteristics, strengths and limitations of SAML, OAuth and ODIC standards. Both SAML and OIDC support a one-time, certificate-based, risk-based, multi-factor, and multi-level authentication and authorization. However, OAuth has been designed for a specific purpose of authorization, consequently, it cannot be the complete solution for any cloud computing model. SAML is an XML-based protocol standard, where XML trees are represented in a verbose form. Every element in the tree has a name (i.e., the element type name), and the element must be enclosed in a matching pair of tags. Whereas, OIDC is a JSON-based protocol standard, and JSON trees are represented in a nested array type of notation similar to that

TABLE I. COMPARISON OF OPEN STANDARD IDENTITY PROTOCOLS: SAML, OAUTH AND OIDC

Criteria	SAML	OAuth	OIDC
1. Authentication	It is a standard for authentication.	It is not a standard for authentication, however, it can accomplish indirect authentication.	It is a standard for authentication.
2. Authorization	It is a standard for authorization.	It is a standard for authorization (delegation) of resources.	It is a standard for authorization.
3. Prime Objective	Federated Identity Management (FIM), Single Sign-On (SSO) for enterprise users.	API authorization between Applications.	Federated Identity Management (FIM), Single Sign-On (SSO) for consumers.
4. Token Format	XML	XML, JSON, JWT	JSON, JWT
5. Token Content	Token contains user identity information but not credentials.	Token contains user identity information but not credentials.	Token contains user identity information but not credentials.
6. Protocol Used	XML, HTTP, SOAP	JSON, HTTP, REST	JSON, HTTP, REST
7. Schemas and Deployments	SPML, SCIM	SCIM	SCIM
8. Roles/Actors	Identity Provider (IDP), Service Provider (SP) and User.	Resource Server (RS), Resource Owner (RO)/User, OAuth Consumer/Client (OC) and Authorization Server (AS).	End User (EU), Relying Party (RP), Authorization Endpoint (AE), Token Endpoint (TE) and UserInfo Endpoint (UIE).
9. Transaction Initiation	SP and IDP initiation.	Consumer/Client (OC) initiation.	Relying Party (RP)/ End User initiation.
10. User Consent	It is not responsible for collecting users consent. However, ECP allows for the exchange of SAML attributes outside the context of a web browser.	It collects users consent before sharing attributes.	It collects users consent before sharing attributes.
11. Claims	No distributed and aggregated claims.	No distributed and aggregated claims.	Distributed and aggregated claims.
12. Client Discovery and On-Boarding	No dynamic introductions.	No dynamic introductions.	Dynamic introductions.
13. Immediate Revocation of Access	It supports revocation. However, in some cases, removal of the user from the identity provider, means that manual suspension must also be performed. Otherwise, authentication will continue using access tokens and SSH keys.	It supports revocation. Token revocation is used to revoke a specified OAuth 2.0 access or refresh token. A revoke token request causes the removal of the client permissions associated with the specified token used to access the user's protected resources.	It supports revocation. Similar to OAuth. However, OIDC has additional ID token that is a cryptographically signed, self-contained token. It allows resource owners to authorize access without a call to the authorization server and it cannot be explicitly revoked.
14. Integrity/ Non-repudiation	XML Signature - X.509; SAML tokens are almost always signed with a private key, as it is a trusted relationship between IDP and SP.	Default bearer token has no proof of possession. However, token contents can be protected by using a DS or a MAC.	JSON Web Signature (JWS)-HMAC SHA-256; [Additional Support -RSA SHA-256 and ECDSA P-256 SHA-256].
15. Confidentiality/ Privacy	XML Encryption-Triple-DES-CBC with 192-bit key and a 64-bit initialization Vector (IV), AES-CBC with a 128-bit initialization vector (IV); [TLS-SSL, Web Services Security (WSS)].	TLS is mandatory to implement with OAuth for token confidentiality. However, token encryption must be applied in addition to the usage of TLS protection.	JSON Web Encryption (JWE)-RSA-PKCS1-1.5 with 2048-bit key, AES-128-CBC, and AES-256-CBC; [Additional Support- ECDH-ES with 256-bit key, AES-128-GCM, and AES-256-GCM].

of Javascript. Consequently, JSON is less verbose than XML, when it is encoded its size is also smaller. This makes OIDC more compact than SAML and a good choice to be passed in HTML and HTTP environments [11]. Both SAML and OIDC support a SSO functionality however, OIDC also provides a user-friendly sign-on approach for mobile and small device based cloud computing. Mobile application development has fewer compatibility issues with OIDC than SAML assertions, and its use is common in mobile apps [11]. OIDC standard works well with both Web browsers and WebViews or native mobile apps. SAML is specially designed for Web browsers. SAML is limited in its ability to support mobile and small devices because of its conceived structure. Perhaps, this may be due to its development time around 2005, when even the first iPhone was not launched. Whereas, OIDC has been designed from the inception to provide services for the web, mobile devices and Internet of Things. It has also been working towards standardising a dedicated version for mobile device called GSMA Mobile Connect standard.

SAML mainly supports the cloud enterprise model (i.e., enterprise-to-enterprise) because its architectural design requires service provider (SP) enterprise and identity provider (IDP) enterprise, and a trusted relationship between them. Whereas, OIDC supports all cloud business models, and which is suitable for both enterprises and consumers. It is the most popular cloud model among consumers and for the untrusted third party association. Communication security is another important parameter in case of open and insecure wireless channels, which is prone to many eavesdropping attacks [12]. The security tokens transported over wireless channels should not be tampered with or altered over its entire life cycle; and its sensitive information should be protected from the disclosure to unauthorized parties. This requires strong digital signature or MAC, and encryption techniques to ensure the integrity and confidentiality of security tokens and its information. Both SAML and OIDC have almost similar security supports. However, signing XML with XML Digital Signature without introducing obscure security holes is very difficult compared to the simplicity of signing JSON [13]. Additionally, JWT does not use **sessions**; therefore, it is free from Cross-Site Request Forgery (CSRF) and many other attacks and offers greater security over SAML token for cloud communications [11].

It has been derived from the comparative analysis and exhibited above that SAML has some issues and requires upgrading to make it suitable for mobile and small resource constrained devices/networks. OAuth is the best fit for the purpose for what it has been developed. However, it is only a delegation protocol, therefore, it has not been developed for full authentication and authorization and cannot be a complete security standard. OAuth is a complementary protocol for both SAML and OIDC. OAuth is already an integral part of OIDC standard, whereas, it can also be embedded into SAML to make SAML more suitable and lightweight for mobile and small device based cloud computing. OpenID Connect would be the best choice for all kinds of cloud computing models as it fulfils most of the requirements for lightweight and secure cloud communications. However, it is an emerging standard, and the final OpenID Connect 1.0 specifications were launched on February 26, 2014 [9]. Furthermore, several leading organisation such as Facebook and Twitter have their own version of OpenID Connect, which are called Facebook and Twitter Connect based on OAuth 2.0. Therefore, OIDC requires more time and enterprise acceptance to become a mature standard for cloud computing.

V. Conclusion

This paper has presented a working prototype and critical analysis of open standard identity protocols SAML, OAuth and OIDC in a cloud computing security paradigm. This critical analysis was based on the evaluation criteria which are explored for this analysis purpose. Subsequently, it has discussed their strengths and weaknesses and determined the most suitable open standard identity protocol for all types cloud computing models. SAML was developed before smart mobile phones and small devices were introduced, and therefore it has many legacy features, which are not compatible with mobile and small devices based cloud computing and would requires upgrading to make it suitable. OAuth is the best fit for the purpose for what it has been developed but not as a complete identity protocol suite. OpenID Connect would be the best choice for all types of cloud computing as it fulfils most of the requirements for it. However, it requires more time and enterprise acceptance to become a mature cloud computing standard identity protocol. In the future, it may be interesting to perform practical investigations on SAML, OAuth and OIDC for various cloud computing models.

References

[1] N. Naik and P. Jenkins, “A secure mobile cloud identity: Criteria for effective identity and access management standards,” in *4th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (IEEE MobileCloud)*, 2016.

[2] E. Bertino, F. Paci, R. Ferrini, and N. Shang, “Privacy-preserving digital identity management for cloud computing.” *IEEE Data Eng. Bull.*, vol. 32, no. 1, pp. 21–27, 2009.

[3] W. Itani, A. Kayssi, and A. Chehab, “Privacy as a service: Privacy-aware data storage and processing in cloud computing architectures,” in *Dependable, Autonomic and Secure Computing, 2009. DASC'09. Eighth IEEE International Conference on*. IEEE, 2009, pp. 711–716.

[4] N. Klingenstein, T. Hardjono, H. Lockhart, and S. Cantor. (2012) OASIS Security Services (SAML) TC. [Online]. Available: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

[5] N. Ranjbar and M. Abdinejadi, “Authentication and Authorization for Mobile Devices,” B.Sc. Dissertation, Department of Computer Science and and Engineering Goteborg, Sweden, 2012.

[6] Pingidentity.com. (2011) A standards-based mobile application idm architecture. [Online]. Available: http://www.enterprisemanagement360.com/wp-content/files_mf/white_paper/exp_final_wp_mobile-application-idm-arch-8-11-v4.pdf

[7] D. Hardt. (2012, October) The OAuth 2.0 authorization framework. [Online]. Available: https://tools.ietf.org/html/rfc6749

[8] R. Boyd, *Getting Started with OAuth 2.0*, 2nd ed. OReilly Media, 2012.

[9] Openid.com. (2014) What is OpenID Connect? [Online]. Available: http://openid.net/connect/

[10] N. Sakimura. (2014) OpenID Connect Core 1.0 incorporating errata set 1. [Online]. Available: http://openid.net/specs/openid-connect-core-1_0.html

[11] JWT.io. (2015) Introduction to JSON web tokens. [Online]. Available: https://jwt.io/introduction/

[12] N. Alsulami and M. M. Monowar, “Data confidentiality and integrity in mobile cloud computing,” *Journal of Emerging Trends in Computing and Information Sciences*, vol. 6, no. 3, pp. 138–143, 2015.

[13] W3.org. (2015) XML security ? issues and requirements. [Online]. Available: http://www.w3.org/2007/xmlsec/ws/papers/09-lockhart-bea/