Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 22 (2013) 680 – 688

Procedia Computer Science

17th International Conference in Knowledge Based and Intelligent Information and Engineering Systems - KES2013

Cloud based Secure and Privacy Enhanced Authentication & Authorization Protocol

Umer Khalid[a], Abdul Ghafoor, Misbah Irum, Muhammad Awais Shibli

[a]National University of Sciences & Technology, H-12, Islamabad 44000, Pakistan

Abstract

Cloud computing is an emerging computing model which facilitates organizations and the IT industry. It helps them to multiply or lessen their resources according to their operational requirements. However, the organizations are reluctant to store their sensitive information on the cloud due to various privacy and identity tracking threats. In the past few years, a lot of research and development efforts have been made to define centralized and federated security mechanisms for the protection of identity information in a cloud environment. However, to the best of our knowledge none of the systems have been designed keeping anonymity as the key component. This paper describes an authentication and authorization protocol which outlines the main features of anonymous communication in the cloud. The solution is an extension of existing standards making it easy to integrate and compatible with existing standards.

Keywords: Cloud Computing, Privacy, Anonymity, XACML, FIPS 196;

1. Introduction

Cloud computing is a general purpose technology that greatly impacts economy in terms of energy, cost and efficiency. It has emerged from distributed computing, evolved incrementally from conceptual grid computing and has been shaped by various other concepts. Organizations such as e-governments, e-learning, hospitals and health care are benefiting from this technology by downsizing ICT setup costs, incorporating multilateral network effects and increasing performance and storage. Moreover, cloud is based on different service models which include the pay-per-service and the pay-per-use service models. These enable organizations to provide low cost flexible growth [1].

Keeping aside the benefits of this technology, organizations now face significant security challenges such as privacy, data leakage and identity protection while migrating applications and sensitive information to and from the cloud [2]. Out of these challenges anonymous communication still needs significant attention and addressing. There are reasons as to why and where anonymity is required. It has been seen in the past that many of the well known cloud service providers have been vulnerable to various kinds of threats including identity theft and data leakage, as discussed in [3] [4] [5].

*Corresponding author. Tel.: +92-333-6377056
E-mail address: 11msccsmkhalid@seecs.edu.pk.

doi:10.1016/j.procs.2013.09.149

Sensitive information stored on the cloud can be replicated and then used for data mining purposes. These key security challenges in cloud need immediate attention from both academia and the industry. A well known fact is that the organizations providing cloud services, such as storage, sell user/customer data for different purposes claiming that they anonymize it before doing so. This claim has been proven wrong in past. Most of this information can be linked with the identity information to leak considerable amount of sensitive data about any organization or individual and cause lack of trust [3] [5].

Different solutions have been proposed to tackle the issues related to identity theft and linkage in cloud out of which the choice of anonymization of identity information is most preferred. Anonymization of identity information provides protection against identity theft and different types of linking attacks. However, the problems associated with the authentication and authorization of anonymized identity information still remains at hand and needs to be further addressed.

One possible solution that has been developed to cater to these problems is anonymous IDMSes. However, with every organization having different security requirements, a cloud service consumer or the provider opting for such an IDMS is often left unclear about the tradeoffs associated with these systems and how these tradeoffs can affect their security requirements [6]. Therefore, after extensively reviewing literature on anonymous authentication and how it works in cloud we have proposed an anonymous authentication and authorization protocol independent of any IDMS.

This paper is arranged in the following order: Section 2 briefly discusses some open standards related to authentication and authorization along with anonymous authentication protocol with the proposed protocol presented in Section 3. Section 4 describes an attack model and the effectiveness of using anonymous identities against such attacks. Conclusion and future line of work is presented in Section 5.

2. Background

Authentication and authorization have been a part of every secure communication system. They have evolved rapidly in order to automate processes and eliminate costs and complexities. Anonymous entity authentication and authorization for cloud are broad and captivating terms often causing misconceptions about the type of services they offer thus requiring further investigation. Different solutions including protocols and identity management systems have existed offering privacy and anonymity some of which have been categorized and discussed. The first step in achieving privacy and anonymity would require trust development as well as efficient management of user identities and user keys. Keeping this in mind we have taken into account some studies related to trust development and key management. We have divided our literature review into two tracks including the modern authentication and authorization protocols such as OAuth, OpenID and Shibboleth along with the traditional anonymous authentication/authorization and trust development mechanisms.

2.1. Existing Solutions

Anonymous authentication generally seems to be a contradictory statement in itself as the task of authentication is proving ones identity while anonymity is to hide ones identity. A number of protocols for anonymous authentication have been proposed based on group, ring signature [7] and public key encryption schemes. Amongst these protocols, the public key based encryption for anonymity enjoys lesser attention.

[8] Suggests construction of an anonymous authentication protocol which is verifiably anonymous. The server encrypts the same challenge along with different random numbers and sends them to the user; the user performs decryption and sends the challenge back to the server. The server upon receiving the value of the challenge from the client compares it to the originally sent value of the challenge and then sends back all the random numbers used to encrypt the challenge. If anyone of the random numbers mismatch the authentication process is unsuccessful.

Such a study is conducted in [9] where the main concept behind authentication is creation of an anonymity set for all the users and then proving their identity by sending N challenges encrypted under N different public keys rather than just a single challenge R. The authors propose verifiable and revocable anonymity allowing this scheme the benefit of ad-hoc features.

Public Key Infrastructure lacks many features when analyzed with respect to the required features in trust management. In [10] the authors analyzed the trust model of PKI along with others to highlight the different shortcomings of these models and proposed a number of features that should be present in an open network.

Trust development and management are major issues related to security in current cloud platforms, therefore analysis of the different trust models in large and distributed environments need to be carried out. Such a study is performed by the authors of [11] after which they propose a novel trust model dividing the resource nodes into domains and sets trust agents thus achieving both identity and behavior authentication on which trust relationship can be built.

Hierarchical cross certification works well for organizations which want their root certification to have control over their subordinate CAs. In case of any other setup such as organizations which require certain amount of cross organizational communication flexibility, peer to peer and bridged certification model is more suited. Such organizations often require development and revocation of trust relationships with the changing policy. An analysis of inter-domain scenarios is done in [12] where the authors show a lab implementation of a scenario where different organizations each having their own CA want to communicate with each other and show that the bridged and peer to peer setup indeed work better.

Health care systems are often linked throughout various entities in order to exchange medical data related to the patients. As of now most of the health information exchange systems provide a unified method of communicating the patient data to different stakeholders such as the patients, the practitioners and the researchers. It is therefore difficult to design a system which provides security and privacy to the sensitive patient data such that information under question still remains interoperable. The authors of [13] bring forth such a framework for exchanging medical information which implements the l-diversity algorithm ensuring that the sensitive medical data is properly anonymized before publicly handed out.

With the above mentioned solutions almost every protocol achieves anonymity using real identity information as the basis for authentication. What makes our work different from existing solutions is that we have proposed a solution which employs identities generated using anonymous certificates [14] and then allocates anonymous identities to the registered users giving our proposed protocol an added layer of anonymity.

2.2. Industrial Solutions

In addition to reviewing the traditional authentication and authorization protocols for achieving anonymity we have also reviewed some open industrial standards for managing centralized and federated identities in cloud environment. Some of these standards are listed below:

2.2.1. OpenID

OpenID is an open standard for user authentication without requiring separate ad hoc systems and helping in consolidation of user identities. It eliminates the need for maintenance of different identities by authentication of the users with different relying parties using a third party service. OpenID provides a framework for communication between the identity provider and the relying party. The user may create OpenID accounts with the identity provider of their choice and then use this identity to authenticate to any service which accepts OpenID authentication. OpenID neither relies on a central authority nor does it mandate a specific mean to authenticate users.Thus, mechanisms as simple as passwords to novel techniques such as smart cards or biometrics can be used for the sake of authentication. OpenID essentially consists of three main entities, an entity wanting to authenticate which is the user, a relying party which wants to verify the user identity and an identity provider which provides the OpenID URLs or XRIs. The authentication takes place with the exchange of the user OpenID identifier which is the XRI or the URL chosen by the user from the web browser [15].

2.2.2. Shibboleth

Similar to OpenID, Shibboleth is the institution centric approach to manage single sign on (SSO) allowing users to authenticate to different services using just one piece of information. It eliminates the need for requirement of different identities and passwords to different services running under federation of different institutes or organizations by signing in users with just one identity and password. Shibboleth is an open source implementation of federated identity based management where the identity providers provide information and the service providers consume this information giving access to content or services [16].

2.2.3. OAuth

OAuth is an open standard for delegating authorization. It provides a solution for clients of any organization to access resources of other clients within or outside the organization. OAuth also provides authorization to any third party to access services and resources of users without sharing any credentials such as passwords or usernames. It is a framework rather than a protocol, so it is interoperable with any newer version of itself [17].

A growing number of social networking websites are now using OAuth for authorization of various applications. Once the permissions are granted the application is able to download the complete stream of social data against the user which is then kept for data mining purposes. By permitting such authorizations OAuth allows the application provider to use it as a tool to facilitate social networking attacks on the users without his or her realization.

3. Construction of an Anonymous Authentication & Authorization Protocol in a Cloud Environment

After a comprehensive review of existing protocols and open industrial standards we designed an architecture consisting of the components essential for providing anonymity as a service in cloud environment. The detailed architecture and its underlying components have been described as follows:

3.1. Designed Architecture

Figure 1 shows the architecture of the protocol needed to achieve authentication and authorization while preserving user anonymity. Like any other traditional solution this architecture contains a strong authentication server, an XACML based policy server for authorization and an IDMS with a key server. However, these components have been used in a novel manner which protects the identity information and has been elaborated in Section 3.

3.1.1. Components

Figure 1 illustrates the main architecture of the protocol described in Section 3, whereas, the components of the architecture at hand have been discussed further in detail below.

3.1.2. IDMS {*Identity Management System*}

The identity management server is responsible for storing all the identity information. It is also responsible for registration of the new users and the distribution of the identity information to all the users. The identity management system consists of the identity store and the key generation and distribution centre. The identity store is responsible for storing all the identities while the key generation and distribution centre is responsible for the generation and distribution of all the user keys.

3.1.3. SA Server {*Strong Authentication Server*}

The authentication server is responsible for authentication of the users. It receives the requests and makes a decision which validates the users as authenticated or unauthenticated. This authentication server is based on strong authentication protocol [18] and verifies the user certificate with the local certificate authority for authentication purposes.

3.1.4. LCA {*Local Certificate Authority*}

The local certificate authority generates and verifies anonymous certificates for the entities wishing to gain access to a certain service. This certificate authority may or may not be verifiable by some other certificate authority in the chain of trust and is dependent on the deployment model. It should be kept in mind that even though the certificate authority is able to verify the user identity based on the certificate, this identity itself is just a pseudonym and not the real identity of the user. The number of anonymous certificate issuing authorities/sub authorities may be further increased or decreased depending on the level of privacy needed.

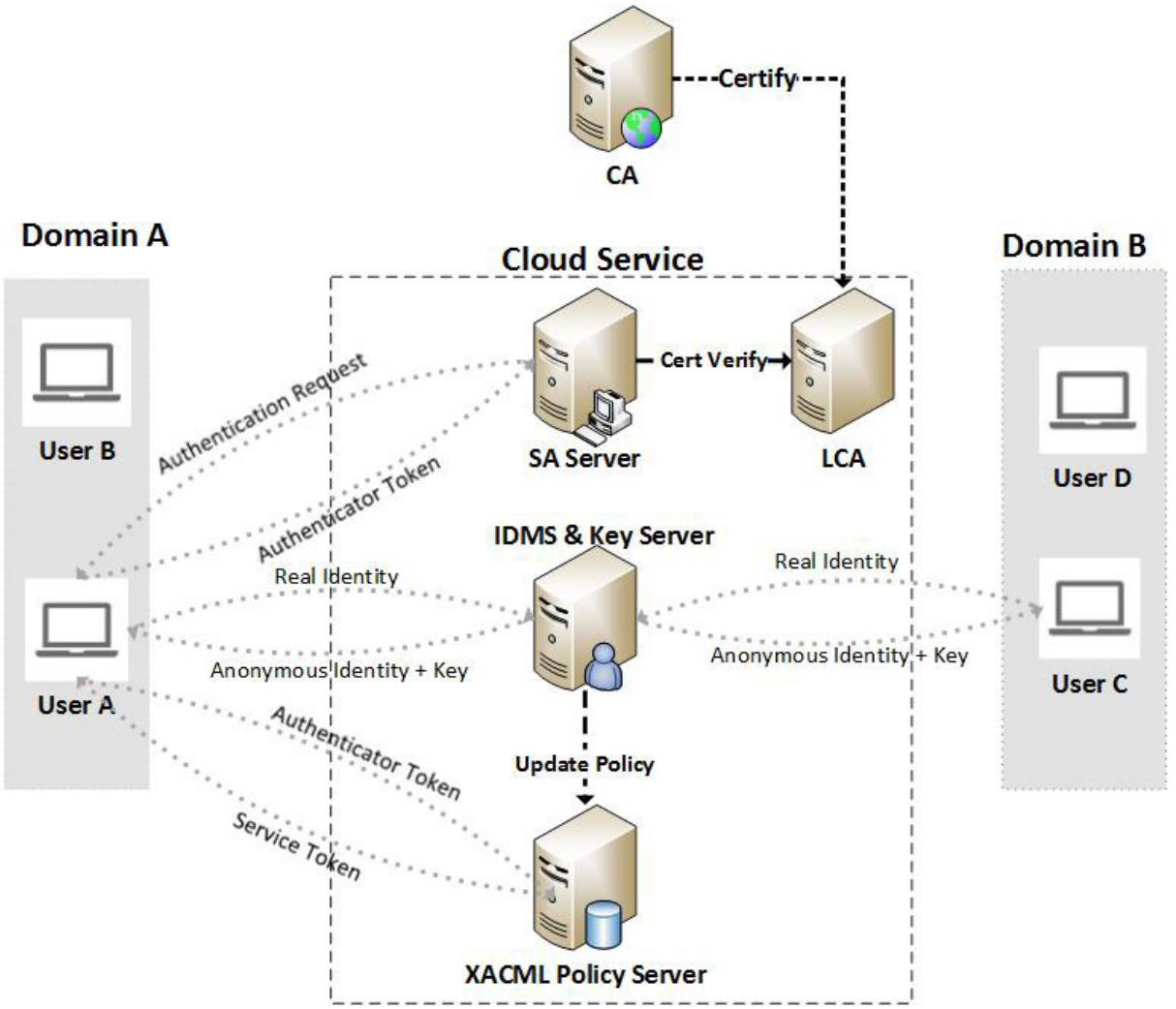

Fig. 1. Proposed Architecture

3.1.5. Authorization Server

The authorization server is responsible for receiving and validating the user access request to some particular service. It maintains a list of all the policies related to the users in the policy engine and updates them when required. If a request is successfully validated, the authorization server issues an access token to the user granting him/her access for a particular amount of time.

3.1.6. Web Application

A web application available to the user through the web browser is responsible for all the interactions of the user with any of above mentioned servers. This web application passes on the requests received from the user to the servers as well as the responses from the server back to the user. In short this web application acts as on demand software granting access to any service.

3.2. Protocol

Before we start defining the anonymous authentication and authorization protocol there are four things that must be considered as pre-requisite.

- Every Client is already registered to the IDMS and the necessary identity attributes are stored in the ID Store.
- All the system components are present on some cloud infrastructure and the client interacts with the web application through the web browser.
- The authentication and the authorization server share a pre-shared master key in order to check the authenticity of the transmitted and the received tokens.
- The described architecture and the solution are given as a cloud service.

3.2.1. Authentication Phase

The Client A begins by selecting a Client B with which it wants to communicate and so builds an authentication request for the SA server. Since Client A cannot directly access Client B without authenticating itself to the SA server first, this request also contains the Client A anonymous certificate.

The *SA Server* receives the authentication request, verifies the certificate with the local certificate authority and the *IDMS*. This is done by sending the *ID* to the*IDMS* which maintains the list of identities in its database. The SA server also sends the certificate to the *LCA* for the verification of its validity.

If both of the verification requests result in success the *SA server* calculates a challenge for the *Client A* which in our case is a *random number*. The *SA server* also attaches an identifier called the *Token ID* which is a *Session Identifier*. The server also stores the generated session and random numbers for future use. If one or both the verification requests either times out or results in a null, the *SA server* terminates.

Client A on receiving the message stores *Session Number* and calculates another random number. *Client A* concatenates the newly generated random number with the received random number, the token id and signs the whole with his/her private key i.e.{RNDA||RNDB||Token ID}Pr_A. It then transmits this message back to *SA Server*. Client A also includes all three values in the unsigned part of the message. The message format looks like: RNDA||RNDB||Tok ID {RNDA||RNDB||Tok ID}Pr_A.

On receiving this message *SA Server* uses the *Client A* Public Key to decrypt the signed part of the message and compares it to the unsigned part as well as the retained value of the random number it sent. If the result is a match then the user gets authenticated. Having authenticated the *Client A*, the *SA Server* generates an ID request. This is a request for assigning a new ID for the client which will be used as the*Anonymous ID* in the future communication.

The *IDMS* generates a new identity, maps the information against the real identity in the identity store and sends back the anonymous identity {AID} to the *SA Server*. *SA Server* sends a request to the authorization server to update the policy against *Client A* with the *AID*. *SA Server* now requests the *KDS or key server* for key creation against the new identity; this key will be used in future while exchanging the group keys with the authorization server. *KDS* generates a new symmetric key for *Client A* and sends it back to the authorization server.

Now possessing an *anonymous identity* and the user key, the *SA Server now* creates an *authentication token* for the user, concatenates it with the received ID and key and encrypts the whole message with the user's public key to safely transmit it to the *Client A*. The format of the authentication token is: Access Token = {AID, RND, Time Stamp/ Expiry Time} E_M. E_M= Symmetric Key shared between the SA server and the authorization server.

3.2.2. Authorization Phase

Client A sends the identities of both the communicating parties i.e. Client As anonymous identity and Client B's real identity along with the access token received from the *authentication server* to the *authorization server*. The *authorization* server after receiving *encrypted access token* and identities stores the IDs and decrypts encrypted access token with the symmetric key to get RND and ID. This proves that the access token was indeed meant for *Client A*. It is to be noted that the token is encrypted using the pre-shared symmetric key shared between the authentication and the authorization server.

The *Authorization Server* checks for policies against both the *anonymous IDs* in its policy engine. It makes a decision based on these policies using the policy combiner algorithm which results in {permit, deny or undefined}. If the result is permit the authorization server sends a key generation request to the *KDS*.

KDS upon receiving the key generation request selects some random users from the groups of 1 and 2 and forms a key ring. This key ring is then further used to generate group key for both 1 and 2. A copy of the

group key is then stored in a field of particular ID in the *IDMS* database while another copy is transmitted to the authorization server. *Authorization server* receives the group key and calculates the service token using the master key {this is the shared master key between the user and the KDS} as follows: Service Token = [GK1] E_{UK1}. It is to be noted that the key UK_1 is the user symmetric key generated at the end of authentication phase.

The *authorization server* transmits E_{UK1} {GK1} to the *Client A*. *Client A* decrypts E_{UK1} {GK1} using hi/her user key {UK1} obtained at the time of authentication and gets GK1. Once it has been authenticated the authorization server similarly transmits the group key GK1 to *Client B* signed with its user key UK2. Both Client A and Client B now start transmitting and receiving data securely. A flow diagram representing the above described protocol is shown in Fig 2.

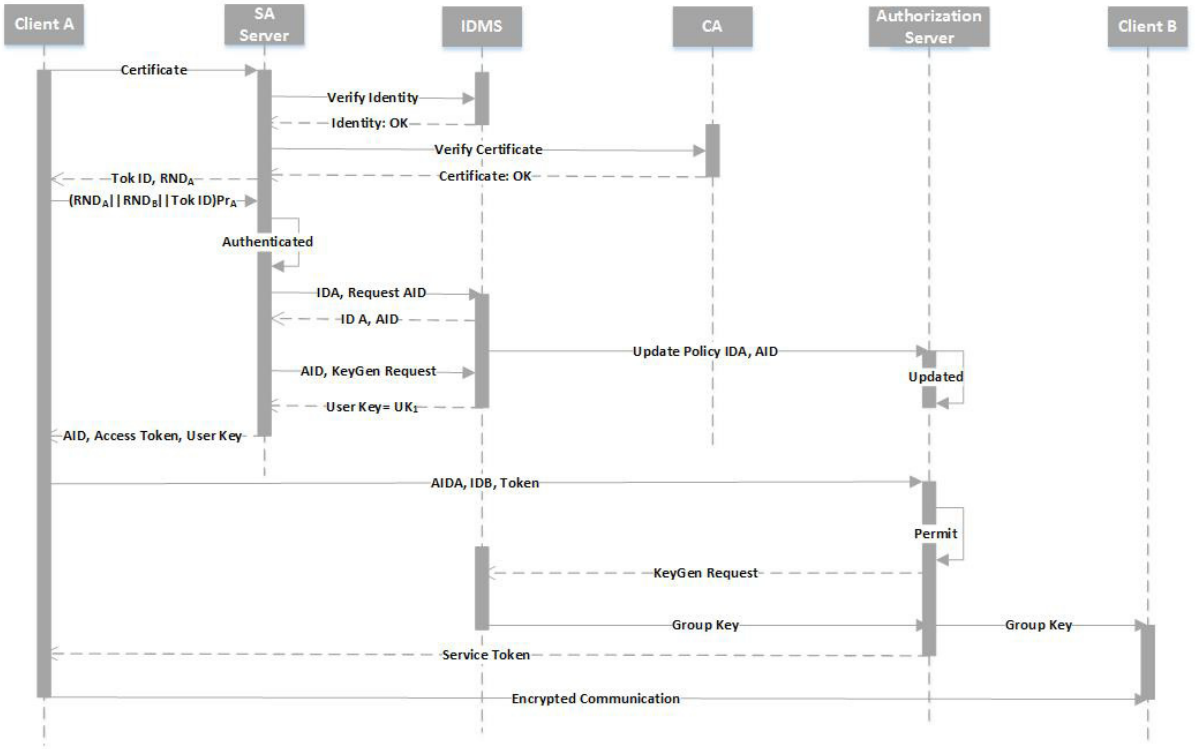

Fig. 2. Protocol Flow Diagram

4. Evaluation

Authors of [19] have analyzed and broadly categorized identity theft related attacks into five categories in application/web applications after a survey. For our protocol we have taken into account all five categories in addition to some others and formulated an attack model to see how these attacks are launched and how our proposed solution for anonymity works against such attacks. These attack types and details are listed in the Table 1.

As future line of work, we plan to extend our protocol by deploying it in a scenario such as health care systems where privacy of the patient data plays a vital role and evaluate its performance using real data sets from such systems. Finally, we plan on implementation, benchmark and a formal security analysis of this protocol.

5. Conclusion

In this paper we designed an anonymous authentication and authorization protocol using anonymous public key certificates along with standard Strong Authentication and XACML servers. The proposed protocol promises

Table 1. Threat Model for Identity related attacks in Web Applications

Attack Types	Launch Scenario	Defensive Measure
Whitewashing	Adversary issues multiple anonymous identities based on one or more than one anonymous certificates.	Issuing limit for identities, typically one anonymous identity per certificate and backtracking anonymous certificate for similar purpose.
SQL Injection	Adversary uses input validation vulnerabilities to send corrupt commands back to the database.	The use of No-SQL databases such as MongoDB along with input white listing.
Phishing	Adversary sets up a fake URL similar to the URL of the real web application convincing the user to enter a valid certificate.	The use of anonymous certificates protects the user identity to be revealed even if the adversary gets hold of a certificate.
Cross-Site Scripting	An adversary inserts a malicious script to a dynamic form presented by the user.	Input sanitization is the answer to this attack, ensuring that the website only returns the input after validating it and filter meta characters while doing so.
Cookie Tampering	Adversary modifies the cookie values sent back to the user web browser.	Use of session identifiers and sequence number instead of transmitting cookies with user credentials.
Session Hijacking	Attacker makes use of the improper implementation session ID number and assumes user identity.	Using anonymous identities after initial authentication and encrypting traffic using symmetric key prevents such sort of attacks.
Remote Web Server Takeover	Attacker makes use of unknown vulnerabilities such as buffer overflow to execute Trojans assuming complete control of the web server.	Prevention of such attacks is only possible by using an IPS with behavior anomaly detection.

full anonymity and prevents identity theft by employing anonymous identities. We have kept our framework flexible enough to provide multiple levels of anonymity by using more than just one CA for issuing anonymous certificates. The protocol has been designed such that the real identity is never used in actual communication and whole process is transparent to the user. Our proposed protocol can be integrated with existing identity management systems and provide anonymity as a cloud service.

Acknowledgements

We would like to express our sincere gratitude to Ms. Rahat Masood for her constant support and guidance. We would like to thank her for motivating us to do our best as well as for her pieces of advice on how we could improve the paper. We would also like to give special thanks to Ms. Anum Farooq for reviewing the paper and providing helpful feedback. We would like to express our appreciation to Ms. Umme Habiba for providing necessary information in Literature review section and also for her support in conducting anonymous IDMS survey. Besides, we would also like to thank KTH-Applied Information Security lab for providing us with a good environment and facilities to complete this paper.

References

[1] A. Bhargav-Spantzel, J. Camenisch, T. Gross, D. Sommer, User centricity: a taxonomy and open issues, Journal of Computer Security 15 (5) (2007) 493–527.
[2] K. Gunjan, G. Sahoo, R. Tiwari, Identity management in cloud computing–a review, International Journal of Engineering 1 (4).
[3] J. Brodkin, Dropbox confirms it got hacked, will offer two-factor authentication.
URL http://arstechnica.com/security/2012/07/dropbox-confirms-it-got-hacked-will-offer-two-factor-authentication
[4] D. Gross, 50 million compromised in evernote hack.
URL http://edition.cnn.com/2013/03/04/tech/web/evernote-hacked
[5] J. Burt, Apple icloud hack raises concerns over cloud security.
URL http://www.eweek.com/c/a/Security/Apple-iCloud-Hack-Raises-Concerns-Over-Cloud-Security-609440/
[6] S. Suriadi, E. Foo, A. Jøsang, A user-centric federated single sign-on system, Journal of Network and Computer Applications 32 (2) (2009) 388–401.
[7] R. L. Rivest, A. Shamir, Y. Tauman, How to leak a secret, in: Advances in CryptologyASIACRYPT 2001, Springer, 2001, pp. 552–565.
[8] Y. Lindell, Anonymous authentication, Journal of Privacy and Confidentiality 2 (2) (2007) 4.

[9] D. Slamanig, Anonymous authentication from public-key encryption revisited, in: Communications and Multimedia Security, Springer, 2011, pp. 247–249.
[10] H. Liping, S. Lei, Research on trust model of pki, in: Intelligent Computation Technology and Automation (ICICTA), 2011 International Conference on, Vol. 1, IEEE, 2011, pp. 232–235.
[11] W. Li, L. Ping, Trust model to enhance security and interoperability of cloud environment, in: Cloud Computing, Springer, 2009, pp. 69–79.
[12] G. López Millán, M. Gil Pérez, G. Martínez Pérez, A. F. Gómez Skarmeta, Pki-based trust management in inter-domain scenarios, Computers & Security 29 (2) (2010) 278–290.
[13] M. Afzal, M. Hussain, M. Ahmad, Z. Anwar, Trusted framework for health information exchange, in: Frontiers of Information Technology (FIT), 2011, IEEE, 2011, pp. 308–313.
[14] N. Zhang, Q. Shi, M. Merabti, Anonymous public-key certificates for anonymous and fair document exchange, IEE Proceedings-Communications 147 (6) (2000) 345–350.
[15] OpenID, It's easy to begin sccepting openid on your website (2013).
URL http://openid.net/add-openid/
[16] S. Cantor, Understanding shibboleth (2010).
URL https://wiki.shibboleth.net/confluence/display/SHIB/UnderstandingShibboleth
[17] E. Hammer-Lahav, Beginners guide to oauth (2010).
URL http://hueniverse.com/oauth/guide/intro/
[18] A. G. Abbasi, S. Muftic, Cryptonet: security management protocols, in: Proceedings of the 9th WSEAS international conference on Data networks, communications, computers, World Scientific and Engineering Academy and Society (WSEAS), 2010, pp. 15–20.
[19] Imperva, The top-5 identity theft attacks.
URL http://www.imperva.com/docs/WP_Top5_Online_Identity_Thefts.pdf