IJCSMA

Prakash Chandra Patra *et al*, International Journal of Computer Science and Mobile Applications,
National Conference on "The Things Services and Applications of Internet of Things",
Gandhi Institute for Education and Technology (GIET) Baniatangi, 23-24 March 2018, pg. 45-55

ISSN: 2321-8363

Impact Factor: 5.515

DENIAL OF SERVICE ATTACKS IN CLOUD-BASED SECURED AUTHENTICATION IN INFORMATION LATTICES AND STANDARD SECURITY REQUIREMENTS

Prakash Chandra Patra[1], Anil Kumar Mishra[2], Tarun Kumar Behera[3]

[1]*Department of Electrical & Electronics Engineering, GIET, Baniatangi, BBSR,* pcpatra@gietbbsr.com
[2]*Department of Computer Science Engineering, GIET, Baniatangi, BBSR,* anilmishra@gietbbsr.com
[3]*Department of Computer Science Engineering, GIET, Baniatangi, BBSR,* tkbehera@gietbbsr.com

Abstract: Now A days there are large number of services based on cloud services, which need identity authentication process very carefully. These services uses gateway which are harmed by most of attackers. Denial of Service which need heavy verification processes consume application service under close look which eliminate risk factors associate with services. Here we propose an authentication protocol suite with consideration of Denial of Service threats having parameter of information lattices and standard security requirement. In this new flexible technique helps to find protocol participation with reliable users only helps the process queue efficient and helps risk of Denial of Service attack minimized.

Keywords: Denial of Service attacks, authentication protocol suite, Federated cloud system standard, Information flow security, Security Authentication Protocol, Cloud computing.

1. Introduction

The importance of cloud computing is rapidly increasing due to the ever increasing demand for internet services and communications. Now a day's Third party service provider owned large server clouds and hosted them both public and private cloud computing resources are used under federated cloud computing (FCC). Presently federated cloud is the deployment and management of multiple cloud computing services with the aim of matching business needs. Data, services, and software are required to be allocated in different clouds for both security and business concerns. Although federated cloud systems (FCSs) can increase the reliability and reduce the cost of computational support to an organization, the large number of services and data on a cloud system creates security risks due to the dynamic movement of the entities between the clouds. As a result, it is necessary to develop tractable formal models faithfully capturing information flow security within FCSs and secure authentication. There are three service layers Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) Mell and Grance, 2011[1]. IaaS provides users with access to physical resources, networks, bandwidth and storage. PaaS builds on IaaS and provides end users access to the operating systems and platforms necessary to build and develop applications, such as databases. SaaS provides end users with access to software applications. Basically Petri nets, colored Petri nets (CPNs) are be used to analyze the correctness of such system. These system detect malicious events which violating the proposed security policy in FCSs. So detecting a Denial of Service attack, in the upper layer (SaaS), is a significant approach to avoid the destruction caused by Denial of Service attacks on the other layers. The service requests for SaaS must be need authentication successfully set of possible access rights as well as lattices of security concern.

The paper is organized as follows. Section 2 provides Related works .Section 3 provides the basic notions about security policies. In Section 4, a model for secure information flow analysis in FCSs is presented. in

Section 5. Discussion with respect problem domain, presents experimental results obtained for the proposed approach. Section 6 concludes the paper.

2. Related work

There are different methods for addressing workflow1 security, the analysis of flow-sensitive programs in Smith (2001) and Russo et al. (2009). Using Petri nets to model workflows, Knorr (2000, 2001) applied the Bell–LaPadula model to workflow security. Knorr (2000) considered the read and write security policies. In Knorr (2001), the deployment of blocks within a workflow across a set of computational resources was not considered. In addition, the paper considered the clearance level but not location level in its embodiment of Bell–LaPadula model. According to Watson (2012) ,in partition workflows over available clouds the security requirements are met in a sophisticated manner,which is based on a multi-level security model that extends Bell–LaPadula to encompass cloud computing. Watson also are very much essential in workflow transformations when data are communicated between clouds. But in dynamic movement of the services and the changes of the clearance level were not under considered. The flow sensitive security model may be capture information flow in FCSs systems, which can be captured by CPNs. However, the clouds and services were assumed to be fixed, and the dynamic movement of services was not considered. In a dynamic environment the location of different classes of data resources as well as users is vital. Therefore Zeng and Koutny (2014) proposed formal model for data resources in a dynamic environment focused on the location. Bell–LaPadula rules and server-side components were not considered.

Gouglidis and Mavridis (2013) proposed a methodology for the development and verification of access control systems in cloud is important. Hence Access control systems against organizational security, which are based on simple transition systems, need to be attentive. So Denial of Service attack at early stage of (SaaS), is significant, to avoid the destruction caused by Denial of Service attacks on the other layers. However, all service requests for SaaS need to be authenticated and must be approved. Saas use OAuth protocol, is widely used authentication protocol that controls the access of third party applications to an HTTP service. Here resource owner can allow a third-party client to access the resources through the owner. After permission to a service (client) to access the client use the resources. The resource server do user authentication. Here the user is authenticated to a server that is trusted by the authorization server, which then issues a credential (such as an access token) to access the resources. There are limitations when the owner shares credentials, such as a username and password, with the third-party client to access restricted resources. The first Limitation is that the access information includes the password, which is most likely stored by a third-party client as clear-text for future access. The second limitation is that the server should only use a password as an authentication method. The third limitation is that the resource's owner cannot limit the access of a third-party client and also cannot control the duration of the access. Finally, if the password is accessible in a third-party client, all of the resources will be accessible, as well. Therefore, OAuth allows a third-party client to access the resources of the server with privileges and rules without using the resource owners' access information. Firstly protocol process starts when the resource owner receives an authorization request from the client. Now resource owner sends back the authorization grant to the client. The client's authorization request determines type of grant namely - Authorization code grants, Implicit grants, Resource owner password credentials grants , Client credentials grants etc. Now client sends an authorization grant and the authentication to an authorization server to obtain the access token. The access token will be provided to the client once the client is authenticated and the authorization grant is validated by the authorization server. The access token replaces authentication, such as username and password, and is also recognized by the resource server. client sends the access token to request restricted resources from the resource server. Server will respond to the request after validating the access token. Authentication protocols everytime lead to vulnerability to Denial of Service attack. Therefore, it is necessary to verify the Denial of Service -resistance in every process of the authentication protocol. Verifying a large number of messages like user credential via the server consumes the resources of the server with more number of degree, particularly when any attacker sends a massive number of forged credential messages as well as Sending typical client credential with each request in the authentication protocol will force the server to verify these requests based on the stored information at the server. As a consequence, the server resources will be exhausted when dealing with a large number of requests.

The protocol is essential to authenticate both the client and the server to each other. Let this protocol uses the ephemeral Diffie Hellman key-exchange where a, b, p, and g are the parameter values of Diffie-Hellman. The protocol, once the server receives a request from a client, the server will begin generating the secret value b. Subsequently, the server will compute the exponential value gb mod p. Moreover, the server will encrypt the nonce of the client and the exponential value via the client public key. Finally, the server will digitally sign the encrypted message. All of these processes will be executed by the server, which consumes a great deal of resources without determining whether the request is legitimate. This mutual authentication, which is vulnerable to Denial of Service attack, is similar to the two-way authentication version of the Transport Layer Security (TLS) protocol (Dierks, 1999).

The strength of the authentication protocol against Denial of Service attacks is basically use cost-based model approach. According to Meadows (2001) mainly demonstrate the effectiveness of the protocols in preventing Denial of Service attacks. The idea behind Meadows approach that one of the participants (the requester or the responder) will get computationally exhausted first. As such, the computation costs for both the requester (client) and the responder (server) need to be determined. The total computation cost of the requester is the total estimated cost of each operation involved in the authentication process on the requester's side during the life of the authentication protocol. However, the total computation cost of the responder is the total estimated cost of each operation during the authentication process until the requester is determined to be either a legitimate requester or attacker.

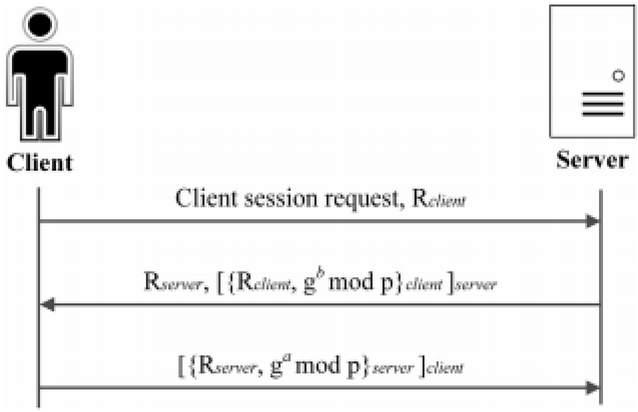
Client
Server
Client session request, R_{client}
R_{server}, $[\{R_{client}, g^b \bmod p\}_{client}]_{server}$
$[\{R_{server}, g^a \bmod p\}_{server}]_{client}$

Figure 1: Mutual authentication protocol

3. Cloud-based secure authentication (CSA) protocol suite

The CSA protocol was developed so that the total computational cost of the client's side will be greater than the resource operations cost of the cloud-based server when they participate in the authentication process together. CSA consists of sets of protocols. The first protocol is used for the registration process, which is an agreement process between the participants (client and cloud server) about certain shared information. Thus, the participants can use that information during the operation of other CSA protocols. The second protocol is an adaptive-based identification protocol that works against Denial of Service attacks. This protocol was developed based on the cost-based model approach. The third protocol is used for the authentication process, which includes all operations that occur based on the initially agreed-upon information of the previous protocols. As a result, cloud server can confirm the identity of the client and then complete the authentication process, or it can detect and then prevent an intruder in the case of a Denial of Service attack. The following parameter

3.1 Security policies in cloud based applications

The concernedness security policies in cloud based applications we emphasizing on two factors as follows

3.1.1 Information lattices

In this paper, we assume that the basis of a federated cloud is a set x of single deployment clouds. More over, S will denote either services, programs and processes, and O will denote objects (e.g., data resources and messages). Subjects and objects will jointly be referred to as entities, and their set will be denoted by E. Now we will assign a security level to any entity, which will in practice be related to the degree of security of its contents, as well as to any cloud which will be related to the maximal security level of the entities it can contain.

Denning and Dorothy 1976, 1982; and Landauer and Redmond 1993 proposed security concern of lattice. Set of Lsec and partial order relation $\leq$sec .so for all L,L' $\in$ Lsec there exists a least upper bound L $\cup$ L' $\in$ Lsec and a greatest lower bound L $\cap$ L' $\in$ Lsec The lattice is complete if each subset L of Lsec has both a least upper bound$\sqcup$ L and a greatest lower bound $\sqcap$L.

3.1.2 Security requirements:

We adopt the Bell–LaPadula multi-level control model of Bell and Lapadula (1973), with services modeled as the subjects S, and data as the objects O (Knorr 2001). Such a security model consists of the following components:

1. A set of possible access rights R.
2. A complete lattice for security concerns.
3. An access control matrix: B : S × O → 2R.
4. A clearance map: c:S → Lsec
5. A security level map: α: E → L sec.

Now system is secure with respect to the above model if the following conditions are satisfied for all subjects' s $\in$ S and objects o $\in$ O:

$$\text{Clearance: } \alpha\,(S\,) \leq_{sec} c(S)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ (1)$$

$$\text{No-read-up: } r \in B(s,o) \Rightarrow c(S) \geq_{sec} \alpha\,(o)\ldots\ldots\ldots\ldots(2)$$

$$\text{No-write-down: } w \in B(s,o)\ \Rightarrow \alpha\,(o) \geq_{sec} \alpha(s)\ldots\ldots.(3)$$

For workflows, the implications of these conditions are that a subject: (i) can only operate at a security level that is less than or equal to its clearance; (ii) cannot read data that is at a higher security level than its own clearance; and (iii) cannot write data residing at a lower security level[20].

As a first step, we extend the model by assigning security levels also to clouds:

$$\alpha: E \cup P \rightarrow L\ sec$$

Moreover, a new mapping loc is used to return the location of each entity:

$$loc: E \rightarrow P$$

Then add an additional rule that an entity can only be deployed in a cloud with a security level that is greater than or equal to that of the entity. That is, for each entity e: an entity e is located in cloud p, then we must have:

- $\alpha\ (loc\ (\ e)\ ) \geq sec\ \alpha\ (e\ )$

3.2 Security level in System model:

We consider capturing the dynamic behavior of federated cloud computing systems and analysis to verify that the system satisfies the requirements, later impose cloud security rule for confidentiality considerations and any user-specified policies. This model uses tuples to represent entities located in the clouds. Each such tuple comprises information about the nature of the entity (service or data), the security information (the security and clearance levels), and the location information (the hosting cloud). Since there can be duplicates of both services and data within a given cloud, the state of the system is a multi set of entities, allowing for an arbitrary multiplicity of any service or object. The transformations of the system are then defined through the simultaneous execution of individual actions, each action being executed instantaneously and possibly many times.

Here we assumed that the system is based on a fixed set of clouds with fixed security levels (issues involved in the modeling of dynamic changes of the set of clouds as well as their security levels are discussed). It is, however, possible to model the dynamic changes of the security levels of subjects and objects as well as their creation and destruction.

(a) Access control sub-system:-
To simplify the presentation, we will assume that a subject can only access a single object at a time. Then, in the access control sub-system, each $\varphi = (\varphi \text{ in } \varphi \text{ out}) \in A$ ac the set of actions Aac is composed of two subsets: the read actions Aac(r) , and write actions Aac(w).

(b) Data flow sub-system:-
Objects can migrate between different clouds. Each action $\varphi = (\varphi \text{ in } \varphi \text{ out}) \in A$ df
A is such that

(c) Control flow sub-system:-
Services can also migrate between different clouds. The last type of actions concerns the migration of the subjects in different locations.

(d) System security:-
We now can capture a key property of information flow across different clouds. That is, a state is secure if all copies of entities present reside in clouds without causing security violation. One can then state a general security policy guaranteeing the security of the system model. Such a policy is formulated by placing a suitable condition on the actions of the model.

After considering above parameter we consider following protocol level.

3.2.1 Registration protocol

In the CSA registration protocol, the client and cloud server will share the required identity data to register the client into the cloud server database. The registration process begins when the client submits all of the required information to cloud server. This information includes the first name, last name, organization name, email address and/or any other information that is required by the cloud service provider. Cloud server will then verify the received information, store it in a database and then send a validation email message to the client to confirm the client's information. After validation, cloud server will activate the client's account. At the same time, cloud server will generate a Unique Encrypted Text (UET) that is encrypted by the cloud server's master key (MK), which is known only by the cloud server. The UET contains client information, such as the Client ID (CID), as well as any other information that will be created by cloud server during the processes of the CSA protocol. The UET is a piece of information that will not be stored on cloud server; rather, it will be sent to the requesting client. Once the client receives the required data from cloud server, both client and cloud server will agree regarding the pre-shared key. The pre-shared key will be created using a key derivation function and a shared secret. The client and cloud server will agree upon the key derivation function and a shared secret at the end of the registration protocol, and they will be exchanged via a secure channel in a very restricted environment. This approach is very much similar to the pre shared key agreement (PSK) used in the UMTS and WPA2 protocols (Southern et al., 2013). Consequently, the client will store the UET and pre-shared key for future authentication processes.

Even if a client is registered to the cloud server, the client cannot access the services available through the cloud server unless cloud server authenticates the client. To perform the authentication protocol that is ready to defend against any internal or external Denial of Service attacks, CSA provides an outer shield to the authentication protocol to help identify the legitimate clients from the Denial of Service attackers. The CSA adaptive-based identification protocol is designed to provide this outer shield in the manner described in the following sub-section.

3.2.2 Adaptive-based identification protocol

The adaptive-based identification protocol utilizes the cost-based model approach, which can be briefly re-stated as follows. Before applying the computational power of the authentication protocols on the server side, the clients are asked to prove their sincere commitment in receiving the cloud server services. This validation of commitment can be achieved by any technique that can force the clients to utilize a significant amount of computational power, before the servers utilize them, to confirm their genuine requests. The adaptive-based identification protocol process functions as follows:

1. Client sends a request for service with a CID to cloud server. At this point, cloud server will block any CID that has performed three consecutive requests within a low time threshold to prevent Denial of Service attacks. The attacker may attempt to launch a Denial of Service attack by sending requests with randomly generated CID values. In this case, the cloud server resources will be less affected than when checking each request for information from a database system because cloud server will simply reply to each request with an verify the subset summation value (S) of the client value.
2. Cloud server will reply directly to the client by sending the puzzle captcha as a challenge, which is the subset summation value (S) along with a cloud server with exactly one time, Rcloudserver. Cloud server will ask the client to prove its commitment regarding receiving the cloud server services by

asking for puzzle captcha solution to the (S) value. The expected solution for this challenge is the vector B (challenge function).

3. Once the client performs a calculation and obtains vector B, the client will send the UET, vector B, the value of S, the R cloud server, the CID and the encrypted timestamp T to cloud server for validation. Note that the notation E(T, Kpreshared) means that the timestamp T is encrypted by the pre-shared key K. At this point, cloud server has all of the information required to validate the authentication requests, so cloud server can apply the validation process to only a few operations, such as (a) By securely hashing and comparing the result vector with the received vector B to determine whether they are similar.(b) To check the time difference between the received encrypted timestamp T and the current time stamp to determine whether it is a reasonable time difference in which to find the solution.

If any of the two previous conditions do not apply, cloud server will drop the request and consider it to be an attacker's request. However, once the client request passes the two conditions, cloud server will decrypt the UET and validate the decrypted information that contains the CID. The participants in the authentication protocol will agree on the session key for future interactions. In addition, they can agree on the sub-session key if they require a refresh process later

3.2.3 Authentication protocol

After the validation process in the previous protocol, cloud server will generate the Session Key (SK), which is encrypted via a pre-shared key Psk. Moreover, cloud server will add both the SK and T information to the UET. Consequently, cloud server is protected against Denial of Service attacks to the storage space because UET will never be saved in the cloud server. Furthermore, cloud server can apply the refreshment property of the session key for future communication by adding the SK to the UET. Now authentication protocol activity as(a) Cloud server will send to the client the generated SK that is encrypted by the pre-shared key Psk, along with the modified UET. (b) Client will confirm the received encrypted SK by sending back the modified UET and the encrypted timestamp T to the cloud server. Therefore, cloud server will decrypt the UET, validate the CID and obtain the SK, then confirm it by decrypting the received timestamp T using the SK. Later, the two parties can agree regarding the sub-session keys by re-applying the processes of the authentication protocol so that the cloud server can generate a sub-session key and add it to the UET without storing it in the cloud system.

4. Analysis of the CSA protocol suite with respect to Information lattices of server and client and Security requirements

Now assessment of the protocol which includes evaluation of the protocol's efficiency against Denial of Service attacks by applying a cost based model approach. The evaluation process measures the computation cost when the client participates in the puzzle captcha solving process during the authentication process.

4.1 CSA protocol suite with cost-based model approach with Information lattices and Security requirement.

Table 1: Validation of the CSA protocol suite in cost-based model approach

Sl	**Client**		**Cloud server**		Information	**Security requirement**
no	**Operation**	**Cost categories**	**Operation**	**Cost categories**	lattices	
1	Request from client	0	Reply directly to the request via secure hashing	0	Least upper bound $\sqcup$ L	Access rights R=1, lattice for security concerns=1, access control matrix 2R=max, clearance map=0, security level α=0.
2	Solve the puzzle captcha until the result is obtained	1	Verify the received input	1/2	least upper bound $L \cup L'$ $\in L_{sec}$	Access rights R=1, lattice for security concerns=1, access control matrix2R=max,clearance map=1, security level α=1.
3	Result send to Cloud server with prevention Denial of Service attacks and decrypted session key	1/2	Decrypt value Generation with encrypted session key ,prevention of Denial of Service attacks	1/2	greatest lower bound $L \cap L'$ $\in L_{sec}$	Access rights R=1, lattice for security concerns=1, access control matrix 2R=max, clearance map=1 , security level α=1.

4.2 Computation cost analysis of solving a subset sum problem with consider Information lattices

We conducted to analyze the time complexity of the subset sum (knapsack) problem. The subset sum can be as follows: given a set of positive integers A of size n and a positive integer value S, does any non-empty subset of size m sum to S. Now A is (10, 74, 20, 60, 3, 56), n = 6, and value of sum S = 94. It is obvious that the subset (74, 20), and value of m = 2, solves the problem because their summation is equal to 94. Finding the binary vector B such that A * B = S solves the problem. So here vector B is the vector (0, 1, 1, 0, 0, 0), and hence A * B = 74 + 20, which is 94. Here identification protocol, the vector B is generated as the output of the Hash function. Here the values of n and m are key factors that play a significant role in the complexity of the subset sum problem. In this experiment, different values of n and m were chosen, for which the time complexity of the subset sum problem was analyzed. The values of n were 128, 256, 512 and >512, while the values of m ranged from 1 to 64. The dynamic programming algorithm is used to solve the puzzle captcha . It is coded in java and run on a 4-core desktop computer with the Windows 10 64-bit operating system, a Core i5-4770 CPU running at 3.90 GHz, and 64 GB of RAM. In our experiment indicates that, when n = 128 and for all values of m, the algorithm solves the puzzle in less than 6 s; when n = 256 and for all values of m, the algorithm solves the puzzle in less than 10 s; and, finally, when n = 512 and for all values of m, the algorithm solves the puzzle in less than 20 s. and, finally, when n >= 625 and for all values of m, the algorithm solves the puzzle in less than 24 s. The detailed execution times when n =625 are shown by the graph in Fig. 2. The graph shows that, when m is equal to 30, the puzzle is solved in approximately 10 s, and, when m was between 50 and 60, the puzzle captcha is solved in approximately 23 s of execution time. It was noticed also that, when n >=512 and m is chosen to be higher than 64, the algorithm slowed down due to the nearly full consummation of the system memory and comprehensively goes to state of system as not responding.

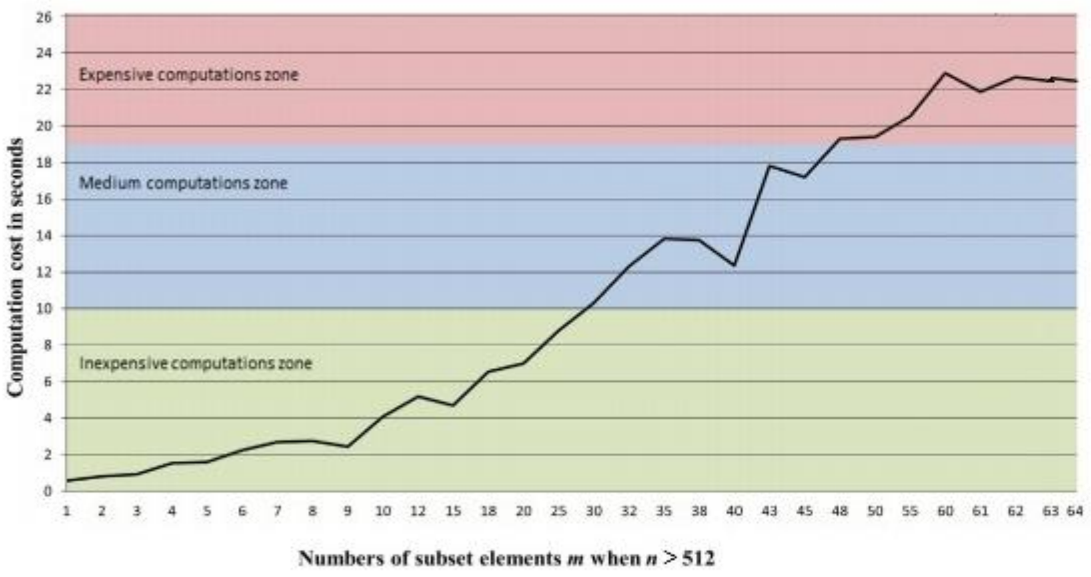

Fig. 2 The execution time of the subset sum problem (in seconds) when n > 512

Based on these above figures and the corresponding system resource consummation, we were able to classify the computation cost of solving the subset sum problem to three main categories: inexpensive=0, medium=1/2 and expensive=1.

5. Discussion

In this work, we have proposed an authentication protocol suite to identify and authenticate cloud users at the SaaS layer[23] and provide a strong shield against Denial of Service attacks. By integrating the client puzzle problem and the utilization of the unique encrypted text (UET), we were able to avoid the security breaches that may lead to Denial of Service attacks. In the CSA protocol suite, we rely on the computational complexity theory to determine different levels of client puzzle captcha solution difficulties. We were thus able to design the identity protocol such that the computational cost incurred by the cloud resources is minimized and the computation cost incurred by cloud users is adjustable based on the service's sensitivity with consideration of Clearance, No-read-up and No-write-down.

The cloud system will not be exhausted because the attack will be detected at an early stage of the authentication process. This protocol suite can be implemented in the SaaS layer of cloud computing systems because the protocol simply relies on basic hardware and software requirements of both the cloud systems and cloud users. Our experiment showed that the traditional software and hardware tools were sufficient to fully implement the protocol and that the dynamic programming algorithm was able to solve the required difficulty levels of the puzzle captcha. Now it needs to be developed for implementation on public or hybrid cloud architectures.

6. Conclusions

In a cloud-computing environment is increasingly common use of software. Verification of users via an authentication protocol is considered to be an initial stage to access these systems. Consequently, the authentication protocol is target of attackers implementing Denial of Service attacks that decreases the availability of cloud services. Using existing strong authentication protocols of traditional network systems in cloud-based applications may lead to Denial of Service attacks vulnerability because the initiation of a massive amount of authentication processes could exhaust the cloud's resources and render the cloud-based application unreachable. In this study, the proposed CSA protocol suite aims to prevent internal and external risks to Denial of Service attacks and number of information lattices. The CSA protocol uses an adaptive challenge technique based on the required efforts of the participants. Using this technique allows the system to identify legitimate requests and pass them to the cloud applications. This CSA protocol suite does not require any external physical device for the authentication process. The effectiveness of the CSA protocol was experimentally analyzed in this work using a cost-based model approach.

References

[1] Mell, P., & Grance, T. (September 2011). The NIST Definition of Cloud Computing. Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, United States Department of Commerce. Gaithersburg, MD 20899-8930: National Institute of Standards and Technology. Retrieved January 28, 2014,

[2] Smith G. A new type system for secure information flow. In: CSFW14. IEEE Computer Society Press; 2001. p. 115–25.

[3] Russo A, Sabelfeld A, Chudnov A. Tracking information flow in dynamic tree structures. In: Proceedings of the 14th European conference on research in computer security. ESORICS'09. Berlin, Heidelberg: Springer-Verlag; 2009. p. 86–103.

[4] Knorr K. 2000. Dynamic access control through Petri net workflows. In: Proceedings of the 16th annual computer security applications conference. ACSAC '00. pp. 159–67.

[5] Knorr K. 2001. Multilevel security and information flow in Petri net workflows. In: 9th International conference on telecommunication systems — modeling and analysis, special session on security aspects of telecommunication systems.

[6] Bell DE, Lapadula LJ. 1973. Secure computer systems: mathematical foundations. Tech. rep., MITRE Technical Report 2547.

[7] Watson P. A multi-level security model for partitioning workflows over federated clouds. J Cloud Comput 2012;1(1):1– 15.

[8] Zeng W, Mu C, Koutny M, Watson P. 2014b. A flow sensitive security model for cloud computing systems. CoRR abs/ 1404.7760.

[9] Choudhury AJ, Kumar P, Sain M, Lim H, Jae-Lee H. A strong user authentication framework for cloud computing. In: IEEE AsiaPacific Services Computing Conference. IEEE; 2011. p. 110-115.

[10] Dierks T. The TLS protocol version 1.0. 1999. http://tools.ietf.org/ html/rfc2246.

[11] Diffie W, Hellman M. New directions in cryptography. IEEE Trans Inf Theory 1976;22(6):644-54.

[12] Hardt D. The OAuth 2.0 authorization framework. 2012. http:// tools.ietf.org/html/rfc6749.

[13] Hwang MS, Chong SK, Chen TY. DoS-resistant ID-based password authentication scheme using smart cards. J Syst Softw 2010;83(1):163-72.

[14]Jaidhar CD. Enhanced mutual authentication scheme for cloud architecture. In: 3rd IEEE International Advance Computing Conference (IACC). IEEE; 2012. p. 70-75.

[15] Juels A, Brainard J. Client puzzles: a cryptographic countermeasure against connection depletion attacks. In: Network and Distributed System Security Symposium (NDSS). Internet Society; 1999. p. 151-65.

[16] Kim M, Fujioka A, Ustaolu B. Strongly secure authenticated key exchange without NAXOS approach. In: Takagi T, Mambo M, editors. Advances in information and computer security. Lecture notes in computer science, vol. 5824. Berlin: Springer Berlin Heidelberg; 2009. p. 174-91.

[17] Lenstra AK, Lenstra Jr HW, Lovasz L. Factoring polynomials with rational coefficients. Math Ann 1982;261(4):515-34.

[18] Meadows C. A cost-based framework for analysis of denial of service in networks. J Comput Secur 2001;9(1e2):143-64

[19] Agarwal A, Madalinski A, Haar S. 2012. Effective verification of weak diagnosability. In: Proc. SAFEPROCESS'12. IFAC. Bell DE, Lapadula LJ. 1973. Secure computer systems: mathematical foundations. Tech. rep., MITRE Technical Report 2547.

[20] Benveniste A, Fabre E, Haar S, Jard C. Diagnosis of asynchronous discrete event systems: a net unfolding approach. Autom Control IEEE Trans 2003;48(5):714–27.

[21] Benzadri Z, Belala F, Bouanaka C. Towards a formal model for cloud computing. In: Service-oriented computing ICSOC 2013 workshops, vol. 8377. Lecture Notes in Computer Science. 2014. p. 381–93.

[22] Clarke EM, Grumberg O, Peled D. Model checking. MIT Press; 1999.

[23] Denning R, Dorothy E. A lattice model of secure information flow. Commun ACM 1976;19:236–43.

[24] Denning R, Dorothy E. Cryptography and data security. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.; 1982.

[25] Germanos V, Haar S, Khomenko V, Schwoon S. 2014. Diagnosability under weak fairness. In: Application of concurrency to system design (ACSD). pp. 132–41.

[26]Germanos V, Haar S, Khomenko V, Schwoon S. Diagnosability under weak fairness. ACM Trans Embed Comput Syst 2015;14(4):1–19.

[27] Gouglidis A, Mavridis I. A methodology for the development and verification of access control systems in cloud computing. In: Collaborative, trusted and privacy-aware e/m-services. Springer; 2013. p. 88–99.